CICS Transaction Server for z/OS
Version 5 Release 3

Performance Guide

SC34-7421-00

CICS Transaction Server for z/OS
Version 5 Release 3

Performance Guide

SC34-7421-00

Note
Before using this information and the product it supports, read the information in “Notices” on page 935.

This edition applies to the IBM CICS Transaction Server for z/OS Version 5 Release 3 (product number 5655-Y04) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

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.

This book does not discuss the performance aspects of the CICS Front End Programming Interface, although it does document the Front End Programming Interface statistics. For more information about the Front End Programming Interface, see the *CICS Front End Programming Interface User's Guide*.

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 many of the books in the CICS Transaction Server 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, see Part 2, "Improving the performance of a CICS system," on page 67.

Location of topics in the Knowledge Center

The topics in this publication can also be found in the CICS Transaction Server for z/OS Knowledge Center. The Knowledge Center uses content types to structure how the information is displayed.

The Knowledge Center content types are generally task-oriented, for example: upgrading, configuring, and installing. Other content types include reference, overview, and scenario or tutorial-based information. The following mapping shows the relationship between topics in this publication and the Knowledge Center content types, with links to the external Knowledge Center:

Table 1. Mapping of PDF topics to Knowledge Center content types. This table lists the relationship between topics in the PDF and topics in the content types in the Knowledge Center

Set of topics in this publication	Location in the Knowledge Center
• Part 1, “Measuring, tuning, and monitoring: the basics,” on page 1 and Part 2, “Improving the performance of a CICS system,” on page 67 • Part 3, “The CICS monitoring facility,” on page 293 • Part 4, “CICS statistics,” on page 411	• Monitoring overview • Improving performance • Monitoring reference in Reference

Notes on terminology

The following abbreviations are used throughout this book:

- “CICS” refers to the CICS element in CICS Transaction Server for z/OS®.
- “MVS™” refers to the operating system, which can be either an element of z/OS or OS/390®.
- “VTAM®” refers to ACF/VTAM.
- “DL/I” refers to the database component of IMS/ESA®.

Changes in CICS Transaction Server for z/OS, Version 5 Release 3

For information about changes that have been made in this release, please refer to *What's New* in the Knowledge Center, or the following publications:

- *CICS Transaction Server for z/OS What's New*
- *CICS Transaction Server for z/OS Upgrading to CICS TS Version 5.3*

Any technical changes that are made to the text after release are indicated by a vertical bar (|) to the left of each new or changed line of information.

Part 1. Measuring, tuning, and monitoring: the basics

Good performance is the achievement of maximizing the use of your system resources, which helps towards reaching service level agreements efficiently.

About this task

You must consider the performance of a CICS system at the following times:
- When you plan to install a new system
- When you review an existing system
- When you plan major changes to a system

The following procedure shows the principal steps to tune a system.

Procedure

1. Agree what good performance is.
2. Set up performance objectives and decide how you measure them.
3. Measure the performance of the production system.
4. Adjust the system as necessary.
5. Continue to monitor the performance of the system and anticipate future constraints.

Chapter 1. Performance monitoring and review

CICS performance can be monitored, measured, and analyzed by implementing a strategy that best suits your needs.

You can use a number of monitoring techniques to set your performance objectives, and analyze CICS performance.

Establishing monitoring activities and techniques

Establishing an ongoing strategy involving monitoring activities and monitoring techniques provides an understanding of your CICS production system that helps to ensure optimum performance and avoid unexpected problems.

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

Monitoring is an ongoing activity for a number of reasons:
- It can establish transaction profiles (that is, workload and volumes) and statistical data for predicting system capacities
- It can give early warning through comparative data to avoid performance problems
- It can measure and validate any tuning you might have done in response to an earlier performance problem.

A performance history database (see “Tivoli Decision Support for z/OS” on page 36 for an example) is a valuable source from which to answer questions on system performance, and to plan further tuning.

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

Strategies include these elements:
- Continuous or periodic summaries of the workload. You can track all transactions or selected representatives.
- Snapshots at normal or peak loads. Monitor peak loads for these reasons:
  - Constraints and slow responses are more pronounced at peak volumes.
  - The current peak load is a good indicator of the future average load.

Procedures, such as good documentation practices, provide a management link between monitoring strategies and tasks.

Tasks (not to be confused with the task component of a CICS transaction) include:
- Running one or more of the tools; see Chapter 2, “Performance measurement tools,” on page 21
- Collating the output
- Examining it for trends

Allocate responsibility for these tasks between operations personnel, programming personnel, and analysts. 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 can 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. Run the tools more frequently at first if required to confirm that the expected peaks correspond with the actual ones.

It is often not practical to keep all the detailed output. File summarized reports with the corresponding CICS statistics, and hold output from the tools for an agreed period, with customary safeguards for its protection.

Do not base conclusions on one or two snapshots of system performance, but rather on data collected at different times over a prolonged period. Emphasise peak loading. Because different tools use different measurement criteria, early measurements might give apparently discrepant results.

Plan your monitoring procedures ahead of time. In your procedures, 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

To collect and analyze data that is consistent with your strategy, you must have the correct tools and processes in place. When you are developing a master plan for monitoring and performance analysis, consider these points:

- Establish a master schedule of monitoring activity. Coordinate monitoring with operations procedures to allow for feedback of online events and instructions for daily or periodic data gathering.
- Consider your business in relation to system performance, for example, what will be the growth of transaction rates and changes in the use of applications and future trends. Consider the effects of nonperformance system problems such as application abends, frequent problems, and excessive attempts.
- Decide on the tools to be used for monitoring. The tools used for data gathering must provide for dynamic monitoring, daily collection of statistics, and more detailed monitoring. See “Planning your monitoring schedule” on page 5 for more information.
- Consider the kinds of analysis to be performed. Take into account any controls you have already established for managing the installation. 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, design worksheets to assist in data extraction and reduction.
- Compose 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 shared with the user liaison group and system performance specialists.
- Create a strategy for implementing changes to the CICS system design resulting from tuning recommendations. Incorporate the recommendations into installation management procedures, and include items such as standards for testing and the permitted frequency of changes to the production environment.

Planning the performance review process

A plan of the performance review process includes a checklist of the tools and analysis that are required to implement monitoring procedures. Establish a simple schedule for monitoring procedures. To create a performance review process, perform the following tasks:

- List 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.
- Create a checklist of review questions.
- Estimate resource usage and system loading for new applications. This is to enable you to set an initial basis from which to start comparisons.

Planning your monitoring schedule

A comprehensive monitoring plan includes the scheduling of various system activities at different time intervals. This approach provides a broad collection of data to measure and analyze the performance your CICS system. Plan for both dynamic monitoring and scheduled monitoring.

Dynamic monitoring

Dynamic monitoring is "on-the-spot" monitoring that you can carry out at all times. This type of monitoring includes the following activities:

- Observing the operation of the system continuously to discover any serious short-term deviation from performance objectives. End-user feedback is essential for this activity. You can also use the Resource Measurement Facility (RMF™) to collect information about processor, channel, coupling facility, and I/O device usage.
- Obtaining status information. You can get status information about system processing during online execution. This information might include the queue levels, active regions, active terminals, and the number and type of conversational transactions. You can get this information with the aid of an automated program started 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 can capture the transaction processing status and measurements of system resource levels.
- Using CICSPlex SM monitoring data. 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. CICS monitoring must be active for CICSPlex SM to collect CICS monitoring information.

Daily monitoring

Measure and record key system parameters by monitoring data daily. The daily monitoring of data usually consists of counts of events and gross level timings. In some cases, the timings are averaged for the entire CICS system. To monitor data daily, perform a series of tasks. For example:

- Record both the daily average and the peak period (usually one hour) average of items such as messages, tasks, processor usage, I/O events, and storage used. Compare these events against your major performance objectives and look for adverse trends.

- List the CICS-provided statistics at the end of every CICS run. Date-stamp 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 as soon as it becomes available.
- Be familiar with all the facilities in CICS for providing statistics at times other than at shutdown. The main facilities 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, including, for example, a shutdown of CICS that causes a gap in the statistics, a complaint from your users of poor response times, a terminal going out of service, or any other significant item. These notes are useful when reconciling disparities in detailed performance figures that might 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 2, "Performance measurement tools," on page 21 for at least part of the day if there is any variation in load. 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.
- Collect CICS statistics, monitoring data, and RMF data into the Tivoli® Decision Support database.

Weekly monitoring

Periodically collect detailed statistics on the operation of your system for comparison with your system-oriented objectives and workload profiles. To monitor data weekly, perform these steps:

- Run the CICS monitoring facility with performance class active, and process it. You might not need to run the facility every day, but it is important to do it regularly and to keep the sorted summary output and the detailed reports. Whether you run the facility on the same day of the week depends on the nature of the system load. For example, if one day of the week has a heavier system load than others, monitor on this day. 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 the load. 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. Checking the batch loads helps you identify any obscure problems because of peak volumes or unusual transaction mixes on specific days of the week. The first few weeks of output from the CICS statistics also provide useful information.You might not need to review the detailed monitor report output every time, but 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 and keep it for an agreed period in case further investigations are required.
- Run RMF, because this shows I/O use, channel use, and other uses. File the summary reports and archive the output information 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 output for more detail.
- Tabulate or produce a graph of values as a summary for future reference.
- Produce weekly Tivoli Decision Support or CICS Performance Analyzer reports.

Monthly monitoring

Monitor and assess trends that are better reflected when tracked regularly over a longer period of time. The following list includes some tasks for monitoring data on a monthly basis:

- 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-stamp 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 might be important. The RMF output provides detailed data on the usage of resources within the system, including processor usage, use of DASD, and paging rates.
- Produce monthly Tivoli Decision Support reports showing long-term trends.

Monitoring for the future

When performance is acceptable, 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 often 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. Identify trends of usage and other factors (such as batch schedules) to aid in this process.

Typical performance review questions

Use the following questions as a basis for your own checklist when carrying out a review of performance data. Many of these questions can be answered by performance reporting packages such as CICS Performance Analyzer or Tivoli Decision Support for z/OS.

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, there might be sign-on errors and, therefore, a lack of terminal operator training. This situation is not a performance problem, but is an example of the additional information that can be provided by monitoring.

1. What are the characteristics of your transaction workload?
   a. Has the frequency of use of each transaction identifier 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 must 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 might 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 CICS Performance Analyzer for z/OS, or Tivoli Decision Support for z/OS.

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, are there any line problems?
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/O operations 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/O operations depending on the queue mix. 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 sync point 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 situation can occur if programs are loaded that then need to open data sets, particularly in IMS™, for example. Can a change in application design rectify the problem?
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 MVS readers and writers, MVS JES, and z/OS Communications Server if running above CICS, and overall I/O, because of the lower-priority regions.
6. What is the coupling facility usage?
   a. What is the average storage usage?
   b. What is the link utilization?
7. Do any figures indicate design, coding, or operational errors?
   a. Are any of the resources heavily used? If so, was this situation expected at design time? If not, can the heavy usage be explained in terms of heavier usage 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? If so, is this occurring 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 could indicate that transactions are being entered without correct operator signon. This situation might indicate that some terminal operators need more training in using the system.

In addition, review regularly certain items in the CICS statistics, such as:
- Times the MAXTASK limit is reached (transaction manager statistics)
- Peak tasks (transaction class statistics)
- Times cushion is released (storage manager statistics)
- Storage violations (storage manager statistics)
- Maximum number of RPLs posted (z/OS Communications Server statistics)
- Short-on-storage count (storage manager statistics)
- Wait on string total (file control statistics)
- Use of DFHSHUNT log streams
- Times auxiliary storage is exhausted (temporary storage statistics)
- Buffer waits (temporary storage statistics)
- Times string wait occurred (temporary storage statistics)
- Times NOSPACE occurred (transient data global statistics)
- Intrapartition buffer waits (transient data global statistics)
- Intrapartition string waits (transient data global statistics)
- Times the MAXSOCKETS limit is reached (TCP/IP statistics)
- Pool thread waits (DB2® connection statistics)

Review the effects of and reasons for system outages and their duration. If there is a series of outages, there might be a common cause.

CICS performance analysis techniques

A number of techniques are available for analyzing CICS performance.

There are four main uses for performance analysis:

- You currently have no performance problems, but you want to adjust the system to give better performance.
- 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.
- A system is departing from previously identified objectives, and you want to find out precisely where and why. Although an online system might operate efficiently when it is installed, the characteristics of the system usage can change and the system might not run so efficiently. This inefficiency can usually be corrected by adjusting various controls. Some adjustments usually need to be made to any new system when it goes live.
- A system might or might not have performance objectives, but it appears to be suffering severe performance problems.

If the current performance does *not* meet your needs, consider tuning the system. To tune your system, you must perform the following tasks:

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 these criteria:
   - Your existing performance objectives
   - Progress so far
   - Tuning effort so far
6. Stop at this point if performance is acceptable; otherwise do one of the following actions:
   - Continue tuning
   - Add suitable hardware capacity
   - Reduce your system performance objectives.

The tuning tasks can be expressed in flowchart form as follows:

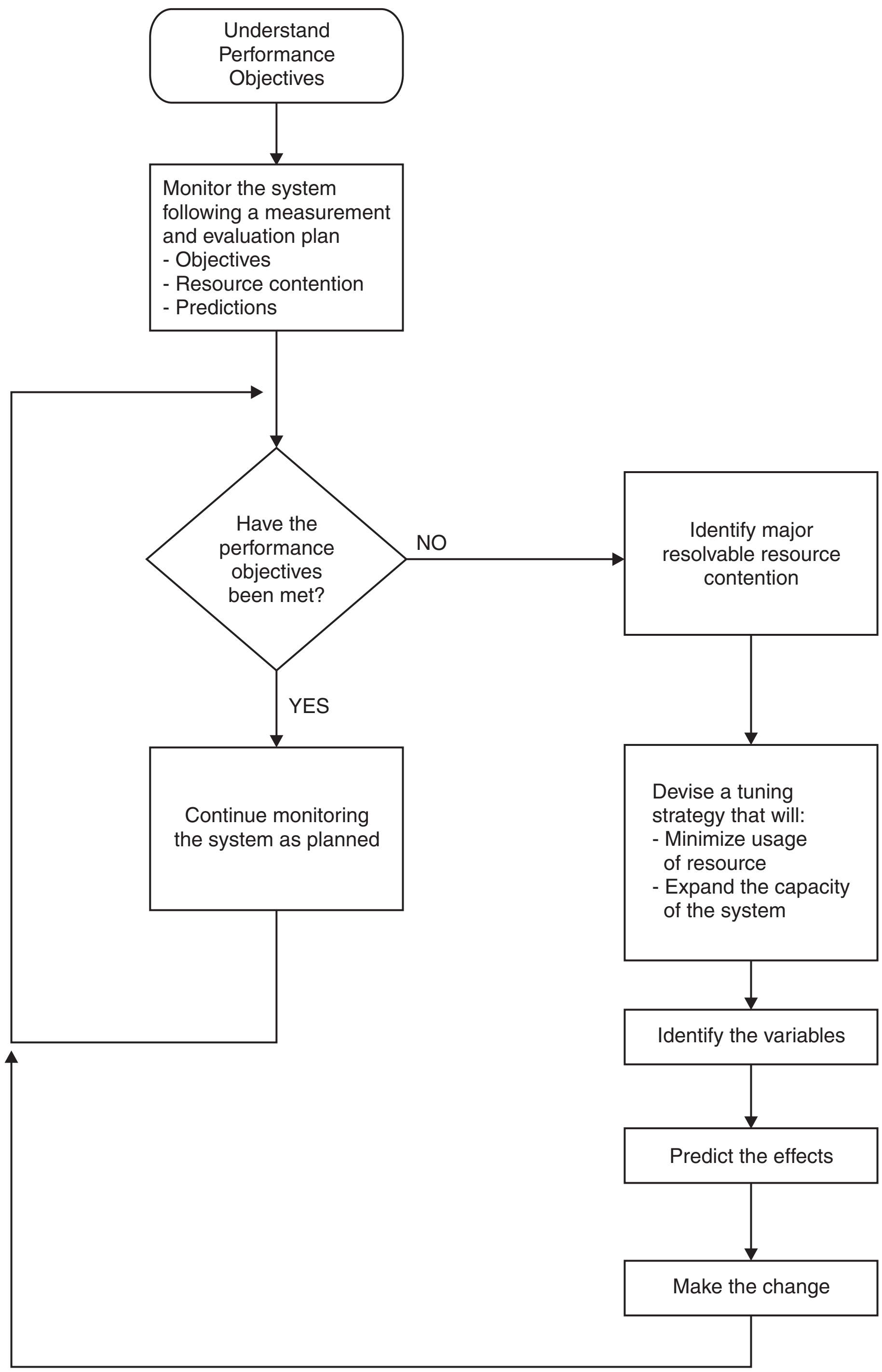


Figure 1. Flowchart to show rules for tuning performance

What to investigate when analyzing performance

Always start by looking at the overall system before you decide that you have a specific CICS problem. Check 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 so, solve the hardware resource problem first. You might 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 that suggest overloads on specific resources. A shorter reporting period can isolate a problem. Consider software and hardware resources; for example, utilization of VSAM strings or database threads, files, and TP lines. Check runtime messages that are 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 the CICS Explorer® and RMF, to monitor the online system and identify activity that correlates to periods of bad performance. Collect CICS monitoring facility history and analyze it, using tools such as CICS Performance Analyzer or Tivoli Decision Support to identify performance and resource usage exceptions and trends. For example, note processor-intensive transactions that perform little or no I/O. These transactions can monopolize the processor, causing erratic response in other transactions with more normally balanced activity profiles. These transactions might 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 small compared with transmission delays and queuing in the network. Use tools such as Tivoli NetView® for z/OS to identify problems and overloads in the network. Without automatic tools, you are dependent on the subjective opinions of a user that performance has deteriorated.

In CICS, the performance problem is either a poor response time or an unexpected and unexplained high use of resources. In general, you must 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, can worsen existing poor performance while it is in use.

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 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 might be a number of possible reasons; for example, the tasks might be trying to do too much work for the system, or the system is real-storage constrained, or many of the CICS tasks are waiting because there is contention for a particular function.

Information sources to help analyze performance

Potentially, any performance measurement tool, including statistics and the CICS monitoring facility, can help in diagnosing problems. Consider each performance tool as usable in some degree for each purpose: monitoring, single-transaction measurement, and problem determination. CICS statistics can reveal heavy use of a particular resource. For example, you might find a large allocation of temporary storage in main storage, a high number of storage control requests per task (perhaps 50 or 100), or high program use counts that imply heavy use of program control LINK.

Both statistics and CICS monitoring might 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 other waits. These waits also generate CICS monitoring facility exception class records.

While these conditions are also evident in CICS auxiliary trace, they might not be obvious, and the other information sources are useful in directing the investigation of the trace data.

In addition, you can gain useful data from the investigation of CICS outages. If there is a series of outages, investigate common links between the outages.

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 performance of the system.

To set up a measurement and evaluation plan, perform the following steps:
1. Devise the plan.
2. Review the plan.
3. Implement the plan.
4. Revise and upgrade the plan as necessary.

To use the plan, perform the following major activities:
- 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 and resource utilization to determine if objectives continue to be met, and whether resources are being used beyond an efficient capacity.
- Keep interested parties informed with 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

Assessing the performance of your system

The following performance measurements can be helpful in determining the performance of a system: processor usage, I/O rates, terminal message or data set record block sizes, paging rates, and error rates.

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 can also increase response time if they are heavily used.

I/O rates
: These rates measure the amount of access to a disk device or data set over a given period. Again, acceptable rates vary depending on the speed of the hardware and response time requirements.

Terminal message or data set record block sizes
: These factors, when combined with I/O rates, provide information about the current load on the network or DASD subsystem.

Indications of internal virtual storage limits
: These indications 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.

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.

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.

Investigate both system conditions and application conditions.

System conditions

A knowledge of the following conditions can help 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

Application conditions, measured both for individual transaction types and for the total system, give you an estimate of the behavior of individual application programs. Gather data for each main transaction, and average values for the total system. This includes the following data:

- Program calls per transaction
- CICS storage GETMAIN and FREEMAIN requests (number and amount)
- Application program and transaction usage
- File control (data set, type of request)
- Terminal control (terminal, number of inputs and outputs)
- Transaction routing (source, target)
- Function shipping (source, target)
- Other CICS requests

Methods of performance analysis

You can use two methods for performance analysis: measuring a system under full production load (*full-load* measurement), to get all information that is measurable only under high system-loading, and measuring single-application transactions (*single-transaction* measurement), during which the system must not carry out any other activities.

Because a system can have various problems, it is not possible to recommend which option to use to investigate the behavior of a system. When in doubt about the extent of a problem, 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.

The IBM Redbooks® publication ABC's of z/OS System Programming, Volume 11 contains further information about performance analysis methods.

Performance analysis: Full-load measurement

A full-load measurement highlights latent problems in the system. It is important that you take 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 might be useful to have the CICS auxiliary trace or RMF active for about 1 minute.

CICS auxiliary trace

CICS auxiliary trace can be used to find situations that occur under full load. For example, all ENQUEUE operations that cannot immediately be honored in application programs result in a suspension of the issuing task. If this situation happens frequently, attempts to control the system by using the master transaction are not effective.

Trace is a heavy overhead. Use trace selectivity options to minimize this overhead.

RMF

It is advisable to do the RMF measurement without any batch activity.

For full-load measurement, the system activity report and the DASD activity report are important.

The most important values for full-load measurement are as follows:
- 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.

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. 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, 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, 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 the 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 might want to move to separate disks those data sets that are periodically frequently accessed.

Comparison charts

Consider using a comparison chart to measure key aspects of your system performance before and after tuning changes have been made. A suggested chart is as follows:

Table 2. Comparison chart

Observations to make		**Run A**	**Run B**	**Run C**	**Run D**
DL/I transactions	Number				
DL/I transactions	Response				
VSAM transactions	Number				
VSAM transactions	Response				
Response times	DL/I				
Response times	VSAM				
Most heavily used transaction	Number				
Most heavily used transaction	Response				
Average-use transaction	Number				
Average-use transaction	Response				
Paging rate	System				
Paging rate	CICS				
DSA virtual storage	Maximum				
DSA virtual storage	Average				
Tasks	Peak				
Tasks	At MXT				
Most heavily used DASD	Response				

Table 2. Comparison chart (continued)

Observations to make		Run A	Run B	Run C	Run D
Most heavily used DASD	Utilization				
Average-use DASD	Response				
Average-use DASD	Utilization				
CPU utilization					

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 items:

- 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*: Identify a selection of terminal-oriented DL/I transactions accessing DL/I databases.
- *VSAM transactions*: 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)*: The RMF paging activity report shows a figure 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 figure is the total paging rate per second for the entire system.
- *Tasks*: 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*: The RMF direct access device activity report, which relates to the most heavily used volume in your system. The “Response” subheading is the figure shown in the “Avg. Resp. Time” column for the volume you have selected. The “Utilization” subheading is the figure shown in the “% Dev. Util.” column for that volume.
- *Average-use DASD*: The RMF direct access device activity report, which relates to a representative average-use volume in your system. The “Response”

subheading is the figure shown in the "Avg. Resp. Time" column for the volume you have selected. The "Utilization" subheading is the figure shown in the "% Dev. Util." column for that volume.

- *Processor utilization*: 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.

Performance analysis: 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 about 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 10th issues 55 SIOs, this results in an average of 10 SIOs per transaction type. This situation should not cause concern if the transactions start at the same time; however, an increase of the transaction rate of the 10th transaction type might lead to poor performance overall.

To investigate this type of problem, you can perform a single-transaction measurement.

Sometimes, response times are good with existing terminals, but adding a few more terminals leads to unacceptable degradation of performance. In this case, the performance problem might be present with the existing terminals, and has been highlighted by the additional load.

To investigate this type of problem, do a full-load measurement and a single-transaction measurement. 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. Halt the polling of remote terminals.

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, insert a pause of several seconds, to make the trace easier to read. Use a copy of the production database or data set for the test, because a test data set containing 100 records can often result in different behavior when compared with a production data set containing 100,000 records.

The condition of data sets can cause performance degradation, especially when many segments or records have been added to a database or data set. Do not measure directly after a reorganization, because the database or data set is only in this condition for a short time. If the measurement reveals an unusually large number of disk accesses, reorganize the data and perform a further measurement to evaluate the effect of the data reorganization.

Single-transaction measurement with only one terminal might not be 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 single-transaction measurement is usually the only means for revealing and rectifying, with justifiable expense, performance degradation under full load.

Ideally, carry out single-transaction measurement during the final test phase of the transactions, for these reasons:

- Any errors in the behavior of transactions can be revealed and rectified before production starts, without loading the production system.
- The application is documented during the measurement phase, helping 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 expected.

If you have many 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, there might be 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, only a few transactions remain, analyze these transactions next, because it is highly probable that these transactions are creating most of the performance problems.

Chapter 2. Performance measurement tools

There are a number of tools that you can use to measure performance and to understand where constraints in the system might develop.

Performance of a production system depends on the utilization of resources such as CPU, real storage, ISC links, coupling facility, and the network. 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, or are supplied as separate products. These topics describe some of the products that can give performance information on different components of a production system.

The list of products in these topics is far from being an exhaustive summary of performance monitoring tools, although 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.

Consider 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 might 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 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.

Often, you need 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% 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:
1. CICS statistics
2. CICS monitoring data
3. CICS internal and auxiliary trace.

Tuning your 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 determining acceptable tuning trade-offs, making tuning changes to your system and reviewing 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 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 and causing a performance degradation.

Making tuning changes to your 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 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.

Reviewing the results of tuning

After each adjustment has been done, review the performance measurements that have been identified as the performance problem to verify that the intended 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.

CICS tools to obtain performance data

You can use CICS statistics, monitoring, and trace facilities to gather and monitor performance data to help you tune your CICS system optimally. Additional sources of information are also listed.

CICS statistics

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

For more information, see Introduction to CICS statistics.

CICS monitoring

CICS monitoring collects data about the performance of all user and CICS transactions during online processing for later offline analysis.

For more information, see Collecting and processing data for CICS monitoring.

CICS trace

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.

CICS trace provides a history of events leading up to a specific situation.

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.

For more information, see Using traces in problem determination in Troubleshooting.

Other sources of information

The measurement tools just described do not provide all the data necessary for a complete evaluation of current system performance. They do not provide information about how, and under what conditions, each resource is being used, or the system configuration that exists when the data is collected. Therefore, it is 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)
- Installed resource definitions
- Link pack area (LPA) map
- Load module cross-reference of the CICS nucleus
- SYS1.PARMLIB listing
- z/OS Workload Manager (WLM) service definition
- MVS System Logger configuration - LOGR couple data set listing
- Dump of the CICS address space
- TCP/IP Profile data set.

System management facility (SMF)

The z/OS system collects statistical data for each task when certain events occur in the life of the task. The System Management Facility (SMF) formats the information that it gathers into system-related (or job-related) records.

System-related SMF records include information about the configuration, paging activity, and workload. Job-related records include information about the processor time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session.

The information gathered by SMF is useful when completing the following tasks:

- Billing users
- Reporting reliability
- Analyzing the configuration
- Scheduling jobs
- Summarizing direct-access volume activity
- Evaluating data set activity
- Profiling system resource use
- Maintaining system security.

For more information, see *z/OS MVS System Management Facility (SMF)*.

Generalized trace facility (GTF)

GTF is part of the MVS system that you can use to record CICS trace entries.

You can use GTF to record CICS trace entries and use the interactive problem control system (IPCS) to produce reports. 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.

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

The amount of processing time that GTF uses 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.

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

You can use the following options to get the data that is generally needed for CICS performance studies:

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

Note: VTAM is now known as z/OS Communications Server.

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 and LOADs, the options are as follows:

```
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, so 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 z/OS MVS Diagnosis Tools and Service Aids.

Generalized trace facility (GTF) reports

You can produce reports from GTF data with the interactive problem control system (IPCS). The reports generated by IPCS are useful in evaluating both system and individual job performance.

IPCS produces job and system summary reports, and also an abbreviated detail trace report. The summary reports include information about 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 other data:
- seek addresses for a specific volume
- arm movement for a specific volume
- references to data sets and members within partitioned data sets
- page faults and module reference in the link pack area (LPA).

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 must 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) shows 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 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 change 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.

CICS Performance Analyzer for z/OS (CICS PA)

CICS Performance Analyzer (CICS PA) is a reporting tool that provides information on the performance of your CICS systems and applications, and helps you tune, manage, and plan your CICS systems effectively.

CICS PA can help:

- System Programmers to track overall CICS system performance and evaluate the results of their system tuning efforts
- Application Programmers to analyze the performance of their applications and the resources they use
- Database Administrators to analyze the usage and performance of database systems such as IMS and DB2
- WebSphere® MQ Administrators to analyze the usage and performance of their WebSphere MQ messaging systems
- Managers to ensure transactions are meeting their required Service Levels and measure trends to help plan future requirements and strategies

CICS PA provides an ISPF menu-driven dialog to help you request and submit your reports and extracts. The available reports and extracts are grouped by category:

- Performance reports
  - List
  - List extended
  - Summary
  - Totals
  - Wait analysis
  - Transaction profiling
  - Cross-system work
  - Transaction group
  - BTS
  - Workload activity

  - Transaction tracking list
  - Transaction tracking summary
- Exception reports
  - List
  - Summary
- Transaction resource usage reports
  - File usage summary
  - Temporary storage usage summary
  - DPL usage summary
  - Transaction resource usage list
- Statistics reports
  - List
  - Alert
  - CICS Transaction Gateway
- Subsystem reports
  - DB2
  - WebSphere MQ
  - OMEGAMON®
- System reports
  - System logger
- Extracts
  - Cross-system work
  - Performance
  - Record selection
  - HDB load
  - System logger
  - Statistics

CICS PA also provides a Historical Database (HDB) facility to help you manage the performance and statistics data for your CICS transactions. SMF data is saved in HDB container data sets that are managed from the CICS PA dialog. The following types of HDB are available:

- Performance List HDB

  A List HDB is built from CMF performance class data. In a List HDB data set, one record represents one transaction. Typically, List HDBs are used to analyze recent transaction events.
- Performance Summary HDB

  A Summary HDB is built from CMF performance class data. In a Summary HDB data set, one record represents a summary of transaction activity over a user-specified time interval. Typically, Summary HDBs are used for long-term trend analysis and capacity planning.
- Statistics HDB

  A Statistics HDB contains collections of CICS statistics and server statistics and CICS Transaction Gateway statistics over a specified time interval.

For more information about CICS Performance Analyzer for z/OS, see the CICS Performance Analyzer documentation.

The CICS PA dialog

You use the CICS PA dialog to create, maintain, and submit your report requests. You can also use it to specify your input data and tailor requests specific to your requirements without needing to understand the CMF data.

The dialog requires no special customization or setup. Reporting can commence immediately.

The following steps explain how to use the dialog for reporting.

1. Define your CICS (and other related) systems and their SMF files and log streams. After your systems are defined, you can start reporting against them. You can fast-track this process by using the take-up facility. CICS PA extracts information about your CICS systems from your SMF files and makes it available in the dialog. If you define your own CMF user fields, specify your MCT definition. The user fields can then be incorporated into your CICS PA reports. The following example panel shows some CICS systems, a DB2 subsystem, a WebSphere MQ subsystem, and an MVS System Logger defined to CICS PA.

```
                              System Definitions                 Row 1 from 8

Command ===> ________________________________________________ Scroll ===> CSR

Select a System to edit its definition, SMF Files and Groups.
                                                                       SMF Files
/   System   Type     Image          Description                       System
_   MVS1     Image             Production MVS system                   MVS1
_   CICSP1   CICS     MVS1     CICS Production System 1                MVS1
_   CICSPTOR CICS     MVS1     CICS Production TOR                     MVS1
_   CICSPAOR CICS     MVS2     CICS Production AOR                     CICSPAOR
_   CICSPFOR CICS     MVS2     CICS Production FOR                     CICSPFOR
_   DB2P     DB2      MVS3     DB2 Production Subsystem                DB2P
_   MQSP     MQ       MVS4     MQ Production Subsystem                 MQSP
_   MVS1LOGR Logger   MVS1     System Logger for MVS1                  MVS1
```

Figure 2. CICS PA: System Definitions

   Related CICS systems, such as those systems that connect through IRC/MRO or ISC/APPC, can be grouped together for reporting purposes. For example, if you assign the CICS MRO systems (CICSPTOR, CICSPAOR, CICSPFOR, CICSPDOR) to a group, you can report on these systems as a single entity. CICS PA reports can then show a complete end-to-end picture of your MRO transaction activity, incorporating detailed DB2 statistics derived from the DB2 accounting data of subsystem DB2P.

2. To build, submit and save your report requests, you can define Report Sets. A Report Set contains the set of reports that you want to run in a single job. Simply select the required reports and submit a report request.

   Figure 3 on page 30 shows a Report Set. The available reports are displayed in a tree structure (folder style) and grouped by category. Report categories can be expanded or collapsed as required. The Active status controls which reports in the Report Set are run when you submit a report request.

```
EDIT                      Report Set - DAILY                   Row 1 of 45
Command ===> ______________________________________________Scroll ===> CSR

Description  . . . . Daily Reports for our production MRO system

Enter "/" to select action.

___          ** Reports **                           Active
     -  ___  Options                                   Yes
             ___  Global                               Yes
     -  ___  Selection Criteria                        Yes
             ___  Performance                          Yes
             ___  Exception                            No
     -  ___  Performance Reports                       Yes
             ___  List                                 Yes
             ___  List Extended                        Yes
             ___  Summary                              Yes
             ___  Totals                               Yes
             ___  Wait Analysis                        No
             ___  Transaction Profiling                No
             ___  Cross-System Work                    No
             ___  Transaction Group                    Yes
             ___  BTS                                  No
             ___  Workload Activity                    No
             ___  Transaction Tracking List            No
             ___  Transaction Tracking Summary         No
     -  ___  Exception Reports                         No
             ___  List                                 No
             ___  Summary                              No
     -  ___  Transaction Resource Usage Reports        No
             ___  File Usage Summary                   No
             ___  Temporary Storage Usage Summary      No
             ___  DPL Usage Summary                    No
             ___  Transaction Resource Usage List      No
     -  ___  Statistics Reports                        No
             ___  Alert                                No
             ___  CICS Transaction Gateway             No
     -  ___  Subsystem Reports                         No
             ___  DB2                                  No
             ___  WebSphere MQ                         No
             ___  OMEGAMON                             No
     -  ___  Performance Graphs                        No
             ___  Transaction Rate                     No
             ___  Transaction Response Time            No
     -  ___  Extracts                                  Yes
             ___  Cross-System Work                    Yes
             ___  Performance                          No
             ___  Record Selection                     No
             ___  HDB Load                             No
             ___  System Logger                        No
             ___  Statistics                           No
             ** End of Reports **
```

Figure 3. CICS PA: Report Set

Report Sets can contain selection criteria, which are used to filter CMF records. This enables you to tailor your reporting to include only the information that you are interested in. For example, you can specify selection criteria to restrict reporting to:

- A particular date/time range
- A group of related transaction IDs
- Transaction response times that exceed your thresholds

3. To tailor the format and content of your reports, you can define Report Forms. You can use an editor to design your own report by selecting the required CMF fields. You can select most CMF fields for reporting, and detailed explanations of each CMF field is available from the dialog. Report Forms can contain

selection criteria. When a report specifies a Report Form and both have selection criteria specified, records must match both sets of selection criteria to be included in the report.

Figure 4 shows a Report Form tailored to show File Control statistics.

```
                     EDIT LIST Report Form - FCLIST       Row 1 of 16 More: >
Command ===> ________________________________________________ Scroll ===> CSR

Description . . . .  File Control List Form          Version (VRM): 700

Selection Criteria:
 _  Performance  *                                   Page width . . 132

   Field
/  Name +       Type     Description
__ TRAN                  Transaction identifier
__ USERID                User ID
__ STOP         TIMET    Task stop time
__ RESPONSE              Transaction response time
__ DISPATCH     TIME     Dispatch time
__ CPU          TIME     CPU time
__ FCWAIT       TIME     File I/O wait time
__ FCAMCT                File access-method requests
__ FCADD                 File ADD requests
__ FCBROWSE              File Browse requests
__ FCDELETE              File DELETE requests
__ FCGET                 File GET requests
__ FCPUT                 File PUT requests
__ FCTOTAL               File Control requests
__ EOR                   ---------------- End of Report ----------------
__ EOX                   ---------------- End of Extract ---------------
```

Figure 4. CICS PA: Report Form

4. Define and maintain Historical Databases (HDBs) as repositories of performance data. Generate reports against your HDBs or export HDB data to DB2 tables for further analysis.

Using CICS PA to analyze CICS performance

CICS PA provides reports and extracts to help you analyze and tune the performance of your CICS systems and applications.

- The Performance List, List Extended, and Summary reports provide a detailed analysis of transaction activity.
- The Performance Totals report provides comprehensive resource usage analysis of your entire CICS system, or individual transactions.
- The Wait Analysis report summarizes transaction activity by wait time. For each transaction ID, the resources that cause this transaction to be suspended are shown in the order of most to least expensive. This report highlights the system resource bottlenecks that might be causing bad response time. More detailed analysis can then be performed, focusing on the problem resources identified.
- The Transaction Profiling report compares two sets of CMF performance class data. For example, you can compare the performance data for a specific CICS application in two different time periods, or the performance data for all applications on two systems.
- The Cross-System Work report combines CMF records from your connected systems (such as MRO and APPC) to produce a consolidated unit-of-work report.
- The Cross-System Work extract consolidates CMF records for the same unit-of-work into a single record in CMF format. CICS PA can then process the

extracted data set to produce any of the reports. For example, “Summarize all multi-system UOWs whose originating transaction ID is TR01”.

- The Transaction Group report provides a detailed list of incoming work requests. Transactions that CICS executes under the same incoming work request (for example, the CWXN and CWBA transactions for CICS web support requests) are grouped together in the report.
- The BTS report provides a detailed list of CICS Business Transaction Services activity. Transactions with the same CICS BTS process identifier (root activity identifier) are grouped together in the report.
- The Workload Activity report provides a transaction response time analysis by z/OS Workload Manager (WLM) service and report class. You can use this information to understand, from a CICS perspective, how well your CICS transactions are meeting their response time goals. The Workload Activity List report is a cross-system report that correlates CMF performance class data from single or multiple CICS systems for each network unit-of-work. Importantly, this report ties MRO and function shipping tasks to their originating task so that their impact on response time can be assessed.
- The Transaction Tracking List report provides performance data for groups of related transactions. This allows monitoring and measurement of transaction performance from the perspective of transaction flow. The report shows how a process flowed from one transaction or system to the next and back again. The report combines CMF records for each originating transaction and its subordinate (group) transactions.
- The Transaction Tracking Summary report combines CMF records for each originating transaction and its subordinate (group) transactions. The summarized data is presented on a single line for each grouped originating transaction.
- The Exception List and Summary reports provide a detailed analysis of the exception events recorded by CMF.
- The Transaction Resource Usage reports process CMF performance data and CMF resource class data to provide a detailed analysis of file, temporary storage, and distributed program link (DPL) usage.
- Statistics Alerts enable you to define conditions, in terms of CICS Transaction Server or CICS Transaction Gateway statistics field values, that interest you. You can then use those conditions to report on CICS statistics stored in SMF files or historical databases.
- The CICS Transaction Gateway reports provide reporting of CICS Transaction Gateway Statistics SMF 111 records. The following reports are available:
  - Activity
  - Usage and Capacity
  - Configuration
  - Client Workload
  - CICS Workload
- The DB2 report processes CICS CMF records and DB2 accounting records to produce a consolidated and detailed view of DB2 usage by your CICS systems. With this report you can view CICS and DB2 resource usage statistics together in a single report. The DB2 List report shows detailed information of DB2 activity for each transaction. The DB2 Summary reports summarize DB2 activity by transaction and program within APPLID.
- The WebSphere MQ report processes WebSphere MQ accounting (SMF 116) records to produce a detailed view of WebSphere MQ usage by your CICS systems. The WebSphere MQ List report provides a trace of WebSphere MQ accounting records. The WebSphere MQ Summary report provides two

summarized views of your WebSphere MQ transactions: by CICS transaction ID showing the WebSphere MQ system and queue resources used, and by WebSphere MQ queue name showing the transactions they service and resources used.

- The OMEGAMON report processes OMEGAMON XE for CICS (SMF 112) records to produce a detailed view of how CICS transactions use Adabas, CA-Datacom, CA-IDMS, and Supra. For each type of DBMS, you can request the following reports:
  - A List report for database usage for each transaction
  - A Transaction Summary report for database usage summarized by transaction ID
  - A Database Summary report for database usage summarized by database
- The System Logger report processes System Logger records to provide information on the System Logger log streams and coupling facility structures that are used by CICS Transaction Server for logging, recovery and backout operations. The report can assist with measuring the effects of tuning changes and identifying log stream or structure performance problems.
- The Performance Graph reports provide a graphical representation of transaction rates and response times.
- The Extract data sets are produced from SMF data and are suitable for further manipulation and analysis. You can use the following extracts to import data into external programs such as DB2, or PC tools such as Lotus® 1-2-3®:
  - A Performance Data Extract for CMF performance class data
  - A System Logger Extract for System Logger data
  - A Statistics Extract for CICS statistics

  The Cross-System Work extract is described earlier in this topic. You can use the Record Selection Extract to reduce the volume of data processed by CICS PA, for more efficient reporting. You can use HDB Load to load SMF data into a Historical Database (HDB).

You can use Report Forms to tailor the format of reports and extracts, for example, to specify which fields, the order of columns, and the sort sequence.

You can use Selection Criteria to filter your reporting, for example to include data for only a specific transaction ID, and only for a specific period of time.

For more information about CICS Performance Analyzer for z/OS, see the CICS Performance Analyzer documentation.

Other tools for obtaining performance data

You can use a number of tools that are not provided by CICS to provide performance-related information to help you optimally tune your CICS system.

The z/OS Resource Measurement Facility™ collects data and produces reports for activity in a sysplex. For more information, see z/OS Resource Measurement Facility (RMF).

The IBM Redbooks publication ABCs of z/OS System Programming contains information about capacity planning, performance management, RMF, and SMF.

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 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 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
- CF activity
- XCF activity.

RMF allows the z/OS 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 can support
- Identify and measure the usage of online channel paths

For more information about RMF, see the IBM Redbooks publication ABCs of z/OS System Programming and z/OS Resource Measurement Facility (RMF) User's Guide.

Tools provided by IMS to obtain performance data

You can use IMS Performance Analyzer (IMS PA) and the IMS program isolation (PI) trace to monitor information on various access methods and other programs used with CICS and the operating system.

IMS Performance Analyzer (IMS PA)

IMS Performance Analyzer is a performance analysis and tuning aid for database and transaction manager systems for IMS. It processes IMS log and monitor data, including fast path data, to provide comprehensive performance, usage, and availability reports that help you to analyze and tune your IMS systems.

IMS PA:

- Uses log and monitor data to produce comprehensive DBCTL reports showing application and internal resource utilization, processor usage, and full function and fast path database activity
- Uses IMS log data to produce comprehensive information about transit times (actual system performance time), and IMS resource usage and availability
- Creates extracts of transit time by time interval data, which can be graphed, exported for processing by external programs, or downloaded to a PC
- Creates extracts of total transaction traffic and exception transactions (MSGQ or fast path), for direct import by external programs
- Processes logs from a single IMS system, or from multiple IMS subsystems running in a sysplex and using shared queues
- Uses monitor data to produce summary and analysis reports for regions, resources, programs, transactions, databases, and the total system, organized by level of detail and area of analysis

For further information, see IMS Performance Analyzer in IBM IMS Tools.

IMS 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. For information about the format of the PI trace report, see *IMS Administration > System administration* in Information Management Software for z/OS product documentation.

TCP/IP monitoring

TCP/IP is a communication protocol used between physically separated computer systems. TCP/IP can be implemented on a wide variety of physical networks. TCP/IP is a large family of protocols that is named after its two most important members, Transmission Control Protocol and Internet Protocol.

Internet Protocol (IP) is a network-layer protocol. It provides a connectionless data transmission service, and supports both TCP and User Datagram Protocol (UDP). Data is transmitted link by link; an end-to-end connection is never set up during the call. The unit of data transmission is the datagram.

Transmission Control Protocol (TCP) is a transport-layer protocol. It provides a connection-oriented data transmission service between applications, that is, a connection is established before data transmission begins. TCP has more error checking that UDP.

UDP is a transport-layer protocol and is an alternative to TCP. It provides a connectionless data transmission service between applications. UDP has less error checking than TCP. If UDP users want to be able to respond to errors, the communicating programs must establish their own protocol for error handling. With high-quality transmission networks, UDP errors are of little concern.

For more information about TCP/IP, see Internet, TCP/IP, and HTTP concepts in Product overview.

You can use TCP/IP management and control to save the data collected by CICS so that it can be examined offline, at some point after the tasks and resources to which it relates are no longer available. You can also use TCP/IP management and control to obtain a CICSplex-wide view of the TCP/IP network and examine items in real time:

- The TCP/IP network resources that a particular CICS region is using.
- The work passing in and out of a particular CICS region over the TCP/IP network.
- The CICS resources and tasks associated with a distributed transaction that flows across the CICSplex over the TCP/IP network.
- The CICS region in which a distributed transaction originated.

You can use TCP/IP management and control to diagnose problems such as connectivity problems and transaction delays, to track work across the CICSplex, to monitor the CICSplex, and to capture system data over time for use in capacity planning.

Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS is an IBM product that collects and analyzes data from CICS and other IBM systems and products.

The reports generated by Tivoli Decision Support are useful when:

- Getting an overview of the system
- Ensuring service levels are maintained
- Ensuring availability
- Performance tuning
- Capacity planning
- Managing change and problems
- Accounting

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

In the reports Tivoli Decision Support uses data from CICS monitoring and statistics. Tivoli Decision Support 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. Tivoli Decision Support for z/OS passes the data and formatting details to Graphic Data Display Manager (GDDM) which does the rest. Tivoli Decision Support 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.

Using Tivoli Decision Support for z/OS 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. Tivoli Decision Support for z/OS 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 counts
- Exceptions and incidents
- Availability.

The following topics describe certain issues and concerns associated with systems management and how you can use the Tivoli Decision Support for z/OS CICS performance feature.

Performance measuring with Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS is a reporting system which uses DB2. You can use it to process utilization and throughput statistics written to log data sets by computer systems. You can use it to analyze and store the data into DB2, and present it in a variety of forms.

Tivoli Decision Support consists of a base product with several optional features that are used in systems management:

- **Tivoli Decision Support for z/OS and optional features:**
- CICS Performance Feature
- IMS Performance Feature
- Network Performance Feature
- System Performance Feature
- Workstation Performance Feature
- iSeries Performance Feature
- Accounting Feature

The Tivoli Decision Support for z/OS 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 DB2 tables
- DB2 table definitions
- Reports.

The Tivoli Decision Support for z/OS 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 nonstandard log data to Tivoli Decision Support for z/OS and report on that data together with data coming from the standard sources.

The Tivoli Decision Support for z/OS CICS performance feature provides reports for your use when analyzing the performance of CICS Transaction Server, based on data from the CICS monitoring facility (CMF) and CICS statistics. These are some of the areas that Tivoli Decision Support can report on:

- Response times
- Resource usage
- Processor usage
- Storage usage
- Volumes and throughput
- CICS and DB2 activity
- Exceptions and incidents
- Data from connected regions, using the unit of work as key
- CICS availability
- CICS resource availability

The Tivoli Decision Support for z/OS CICS performance feature collects only the data required to meet CICS users' requirements. You can combine that data with more data (called *environment data*), and present it in a variety of reports. Tivoli Decision Support for z/OS provides an administration dialog for maintaining environment data. Figure 5 on page 39 illustrates how data is organized for presentation in Tivoli Decision Support z/OS reports.

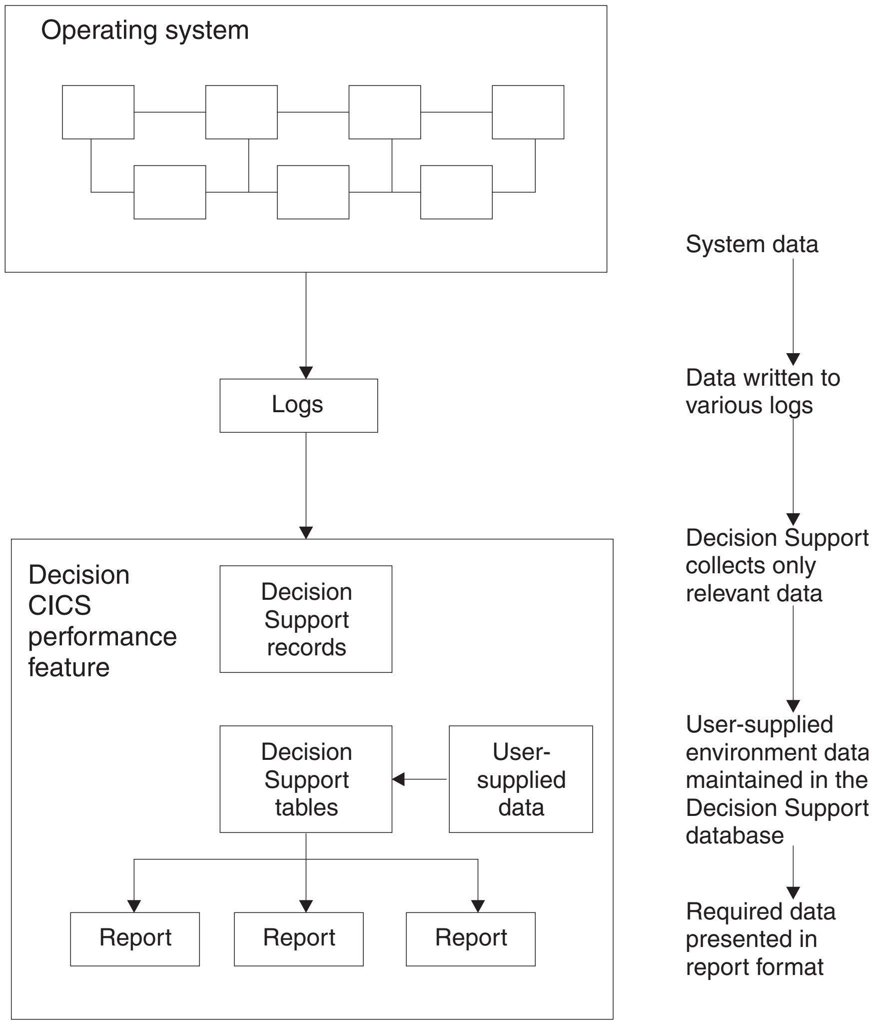


Figure 5. Organizing and presenting system performance data

The Tivoli Decision Support for z/OS CICS performance feature processes these records:

CMF

- CICS Transaction Server performance
- CICS Transaction Server exceptions
- CICS Transaction Server accounting, performance, and exceptions

Statistics

- CICS Transaction Server statistics

Monitoring response time:

The response time is the total time from the start to the finish of the transaction's activity, subdivided into suspend time and dispatch time. The dispatch time includes service time. You can use the Tivoli Decision Support for z/OS CICS response-time reports to see the CICS application internal response times.

The elements of the response time report are shown in Figure 6 on page 40.

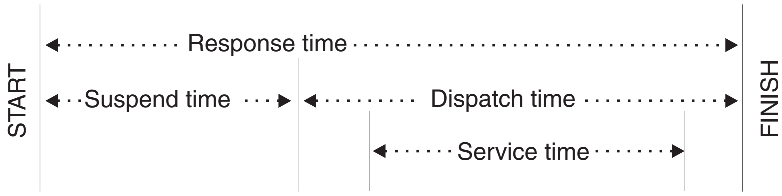


Figure 6. CICS internal response-time elements

As described in *Decision Support Network Performance Feature Reports*, the Network Performance feature generates reports that show the total, end-to-end average response time (operator transit time) for SNA 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 the Tivoli Decision Support for z/OS 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). Tivoli Decision Support-supplied reports can help you isolate a resource as the cause of a CICS performance problem.

If both the Tivoli Decision Support for z/OS CICS performance feature's statistics component and the Decision Support 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.

Tivoli Decision Support for z/OS produces several reports that can help analyze storage usage. For example, the CICS Dynamic Storage (DSA) Usage report, shows pagepool usage, under the headings **Pagepool name**, **DSA (bytes)**, **Cushion (bytes)**, **Free storage (bytes)**, **Free storage (pct)**, **Largest free area**, **Getmains**, and **Freemains**.

```
          CICS Dynamic Storage (DSA) Usage
        MVS ID ='MV28'      CICS ID ='IYCSCTSK'
         Date:  '2001-01-17' to '2001-01-18'
```

Pagepool name	DSA (bytes)	Cushion (bytes)	Free storage (bytes)	Free storage (pct)	Largest free area	Getmains	Freemains
CDSA	524288	65536	299008	57	245760	2668	2470
ECDSA	5242880	131072	1122304	21	868352	1084154	1067000
ERDSA	11534336	262144	1130496	9	966656	710	16
ESDSA	0	0	0	0	0	0	0
EUDSA	2097152	0	2097152	100	1048576	73620	73620
RDSA	524288	65536	204800	39	122880	40	0
SDSA	262114	65536	249856	95	249856	12	6
UDSA	524288	65536	524288	100	262114	301922	301922

Tivoli Decision Support Report: CICS809

Figure 7. CICS Dynamic storage (DSA) usage report

Monitoring volumes and throughput:

If you suspect that a performance problem is related to excessive paging, you can use Tivoli Decision Support for z/OS to report on page-ins, using RMF data.

Because CICS Transaction Server for z/OS, Version 5 Release 3 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.

The best indicator of a transaction's performance is its response. For each transaction ID, the CICS transaction performance detail report (in Figure 8 on page 42) shows the total transaction count and the average response time. The headings are Tran ID, Tran count, Average resp time (sec), Average CPU time (sec), Prog load reqs (avg), FC calls (avg), Exceptions, Program storage bytes (max), Getmains < 16 MB (avg), and Getmains > 16 MB (avg). 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 counts 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.

```
                   CICS Transaction Performance, Detail
                    MVS ID ='MV28'  CICS ID ='IYCSCTSK'
                   Date: '2001-01-17'  to  '2001-01-18'


                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

                    Tivoli Decision Support Report: CICS101
```

Figure 8. CICS transaction performance, detail report

Combining CICS and DB2 performance data:

You can create reports that show the DB2 activity caused by a CICS transaction by 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 9 shows how DB2 data can be correlated with CICS performance data using the DB2 token (QWHCTOKN) to identify the task.

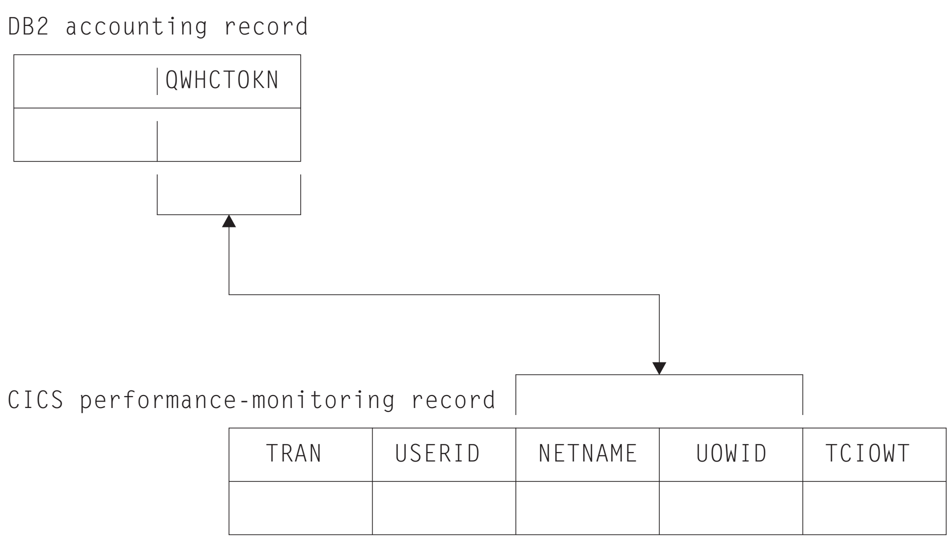


Figure 9. Correlating a CICS performance-monitoring record with a DB2 accounting record

Matching the NETUOWPX and NETUOWSX 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 Tivoli Decision Support for z/OS 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
- z/OS Communications Server request rejections
- I/O errors on auxiliary temporary storage
- I/O errors on the intrapartition transient data set
- Autoinstall errors
- MXT reached
- Link errors for IRC and ISC
- Log stream buffer-full conditions
- CREAD and CWRITE fails (data space problems)
- Local shared resource (LSR) pool (string waits)
- Waits for a buffer in the LSR pool
- Errors writing to SMF
- No space on transient-data data set
- Waits for a transient-data string
- Waits for a transient-data buffer
- Transaction restarts
- Maximum number of tasks in a transaction class reached (CMXT)
- Transmission errors

Figure 10 on page 44 shows an example of an incidents report, giving information on Severity, Date, Time, Terminal operator ID, User ID, Exception ID, and Exception description.

```
                               CICS Incidents
                        DATE: '2001-01-17' to '2001-01-18'

                      Terminal
                      operator  User    Exception          Exception
Sev  Date       Time      ID     ID       ID               description
---  ---------- -------- -------- -------- ------------------ ---------------------------
03   2001-01-17 15.42.03 SYSTEM            TRANSACTION_ABEND  CICS TRANSACTION ABEND AZTS
03   2001-01-18 00.00.00 SYSTEM            TRANSACTION_ABEND  CICS TRANSACTION ABEND APCT
03   2001-01-18 17.37.28 SYSTEM            SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL
03   2001-01-18 17.45.03 SYSTEM            SHORT_OF_STORAGE   CICS SOS IN PAGEPOOL

                Tivoli Decision Support report: CICS002
```

Figure 10. Example of a Tivoli Decision Support CICS incidents report

Tivoli Decision Support for z/OS 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 from one region (or processor complex) to another and back. Using the data from the trace, you can determine the total resource requirements of the combined transaction as a unit of work, without separately analyzing 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 11 on page 45 shows an example of how Tivoli Decision Support for z/OS presents CICS unit- of-work response times. The headings are Adjusted UOW start time, Tran ID, CICS ID, Program name, UOW tran count, and Response time (sec).

```
                     CICS UOW Response Times
                 Time: '09.59.00' to '10.00.00'
                        Date: 2001-01-18


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  DFHMIRS
          ARUS  CICSPRDC  AR49000

09.59.27  CSM5  CICSPRDB  DFHMIRS        4     0.639
          CSM5  CICSPRDB  DFHMIRS
          MQ01  CICSPROD  DFHAPRT
          MQ01  CICSPRDD  CMQ001

 ...

        Tivoli Decision Support report: CICS902
```

Figure 11. Tivoli Decision Support for z/OS CICS UOW response times report

Monitoring availability:

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.

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, telecommunication 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.

Tivoli Decision Support for z/OS can help you, because all the data is in one database.

CICS workload activity reporting:

CICS records the transaction ID, the associated terminal ID, and the elapsed time at the end of each transaction. When more detailed reports are needed, Use the MVS Performance Management (MVSPM) component of System Performance feature of Tivoli Decision Support.

Transaction data is useful when you require only transaction statistics, rather than the detailed information that CMF produces. In many cases, it is 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 MVSPM component of the System Performance feature of Tivoli Decision Support 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 Tivoli Decision Support reports for CICS in this environment. Figure 12 shows how service classes were served by other service classes. This report is available only when the MVS system is running in goal mode. The headings are Workload group, Service class, Served class, No of times served, No of transactions, and No of times served per transaction.

```
                  MVSPM Served Service Classes, Overview
                  Sysplex: 'SYSPLEX1' System: IP02
                  Date: '2001-01-18' 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

                  Tivoli Decision Support report: MVSPM79
```

Figure 12. Example of an MVS Performance Management served service classes overview report

Figure 13 on page 47 shows the average transaction response time trend and how the various transaction states contribute to it. (The times shown for the various transaction states are calculated based on transaction state samples, and so are not necessarily a precise record of the time spent in each state.) Adding the time spent in each of the transaction states (the shaded areas on the graph) gives the average execution time, which is shorter than the average response time (the line on the graph). 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.

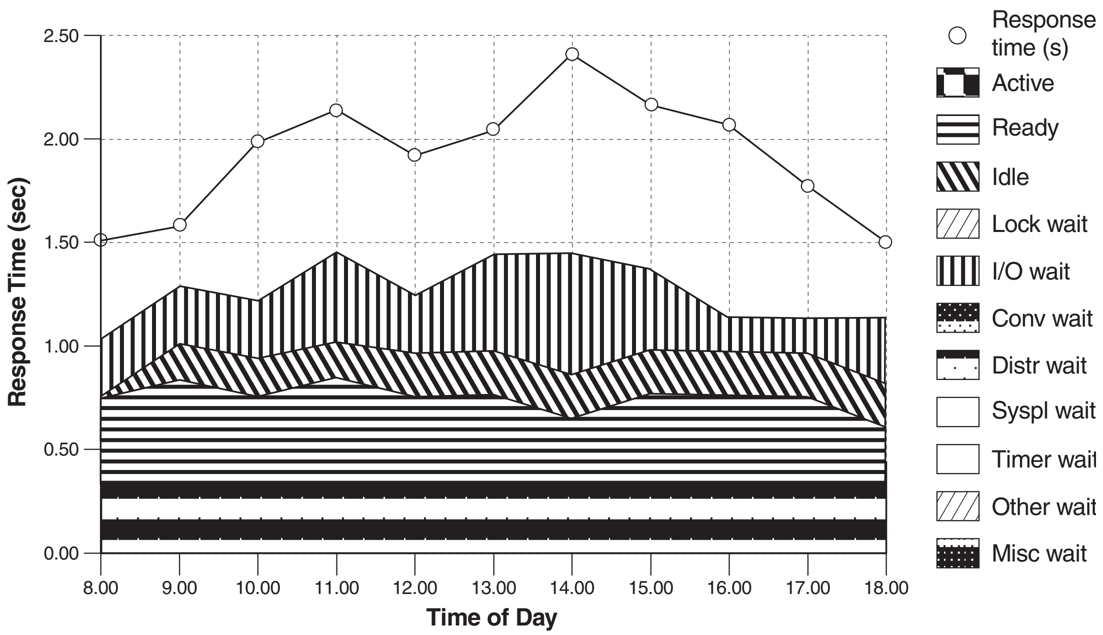


Figure 13. Example of an MVS Performance Management response time breakdown, hourly trend report

Figure 14 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. The report gives information on Workload group, Service class/Period, Ph, MVS sys ID, and Total state, followed by the percentage of response time spent in each of the states listed in Figure 13.

MVSPM Response Time Breakdown, Overview
Sysplex: 'SYSPLEX1' Subsystem: IP02
Date: '2001-01-18' Period: 'PRIME'

Workload group	Service class /Period	Ph	MVS sys ID	Total state (%)	Activ state (%)	Ready state (%)	Idle state (%)	Lock wait (%)	I/O wait (%)	Conv wait (%)	Distr wait (%)	Local wait (%)	Netw wait (%)	Syspl wait (%)	Timer wait (%)	Other wait (%)	Misc wait (%)
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

Tivoli Decision Support report: MVSPM73

Figure 14. Example of an MVS Performance Management response time breakdown overview report

Tivoli OMEGAMON XE for CICS on z/OS

Tivoli OMEGAMON XE for CICS on z/OS helps you to proactively manage performance and availability of complex CICS systems.

Tivoli OMEGAMON XE for CICS on z/OS (OMEGAMON XE for CICS on z/OS) is a remote monitoring agent that runs on z/OS managed systems. It assists you in anticipating performance problems and warns you when critical events take place in your CICS environments. You can set threshold levels and flags to alert you when events within your CICS regions reach critical points.

When running under the Tivoli Enterprise Portal, IBM Tivoli OMEGAMON XE for CICS on z/OS offers a central point of management for CICS Transaction Server and provides a comprehensive means for gathering the information you need to detect and prevent problems within your CICS regions. You view data that Tivoli Enterprise Portal gathers in tables and charts that show you the status of your managed CICS regions.

With this data you can perform a number of tasks:

- Collect and analyze reliable, up-to-the-second data that allows you to make faster, better informed, operating decisions
- Manage all CICS regions from a single point to identify problems at any time
- Balance workloads across various regions
- Track performance against goals

With OMEGAMON XE for CICS on z/OS, systems administrators can set threshold levels and flags to alert them when system conditions reach these thresholds. These are the advanced monitoring facilities:

- User-defined and predefined situations based on thresholds to raise different types of alerts
- At-a-glance status of all CICS regions
- The capability to monitor multiple CICS regions simultaneously from one or more centralized workstations

Used in conjunction with other OMEGAMON XE monitoring products, the data, analyses, and alerts presented by OMEGAMON XE for CICS on z/OS help you develop a holistic view of your entire computing enterprise from a single console.

OMEGAMON XE for DB2

Tivoli OMEGAMON XE for DB2 Performance Expert on z/OS is a single, comprehensive assessment tool, Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS helps you resolve critical performance issues.

Performance Expert

OMEGAMON XE for DB2 Performance Expert (PE) is a performance analysis, monitoring, and tuning tool for DB2 on z/OS environments. This product is part of the integrated and complete cross z Systems™ monitoring solution of the IBM Tivoli OMEGAMON XE family that monitors all DB2 subsystems on z/OS and other resources, such as IMS, MVS, or CICS. OMEGAMON XE for DB2 PE simplifies and supports system and application performance monitoring, reporting, trend analysis, charge back usage, and buffer pool analysis. If problems are encountered you are notified and advised how to continue.

Performance Monitor

Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS permits you to monitor, analyze and optimize the performance of DB2 on z/OS applications in two key modes: online, in real time with immediate alerts when problems occur, and batch, in reports.

Tivoli OMEGAMON XE for DB2 Performance Monitor on z/OS helps you resolve critical performance issues. Use it to monitor:

- Individual data-sharing members or entire data-sharing groups.
- Applications running in a parallel query environment, even in the parallel tasks are executed on different processors.
- Near-term performance history to see problems that otherwise go unnoticed and prevent them in the future.
- Object analysis of database disks, tables, table spaces, and other elements to tune performance.

For more information about Tivoli OMEGAMON XE for DB2 on z/OS, see IBM Tivoli Monitoring Information Center.

Chapter 3. Identifying CICS performance constraints

Major constraints on a CICS system are often identified by external symptoms such as stress conditions and longer response times. CICS can resolve some constraint problems; others must be resolved manually.

Many indications of poor performance can occur in a system that is congested. For example, if there is a slowdown in direct access storage device (DASD) activity, the following symptoms might occur:
- Transactions that perform data set activity accumulate
- Waits on strings occur
- More transactions are waiting in the system
- Demands on virtual storage increase
- Demands on real storage increase
- Increased paging occurs
- The task dispatcher uses more processor power scanning task chains
- Task constraints occur
- The MXT or transaction class limit is exceeded; the processor is required to do additional work because more retries are required

As a result the system shows heavy use of all resources, resulting in typical system stress. This situation does not indicate problems with all resources; it shows that a constraint has yet to be found. To identify the constraint, you must find out what is affecting task life.

If performance is unacceptable, the performance constraints (the causes of the symptoms) must be identified so that they can be tuned.

When dealing with limit conditions, you might find it helpful to check the various hardware and software locations in the system where performance constraints are occurring.

Hardware contentions

Contentions can occur on processor cycles, real storage, database associated hardware I/O operations, and network-associated hardware operations.
- *Processor cycles*. It is not uncommon for transactions to execute more than one million instructions. To execute these instructions, transactions must contend with other tasks and jobs in the system. Sometimes these tasks and jobs must wait for activities such 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 determine which transactions or jobs get use of the processor, and batch or other online systems affect response time by receiving preferential access to the processor. Batch programs that access 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.
- *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

that occurs when acquiring 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).

- *Database-associated hardware (I/O) operations*. 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.

  Large amounts of central and expanded storage, very large data buffers, and keeping programs in storage, can significantly reduce DB I/O contention and somewhat reduce processor utilization while delivering significant internal response time benefits.
- *Network-associated hardware operations*. 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 also affect performance. 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.

Design considerations

The length of time between data set reorganizations can affect performance. The efficiency of access decreases as the data set becomes increasingly fragmented. Fragmentation can be kept to the minimum by reducing the length of time between data set reorganizations.

The following factors can limit performance:

- *Database design*. A data set or database needs to be designed to meet 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.
- *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-duplex or half-duplex capabilities can affect performance.

- *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 any one time, and this can also cause a performance constraint. Examples of concurrent processes include the SNA 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 SNA 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 the z/OS Communications Server for SNA.

At an SNA LU enter the following:

```
IBMTEST (n)(,data)
```

where *n* is the number of times you want the data echoed, and *data* consists 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 SNA LU.

IBMTEST is an echo test designed to give the user a rough idea of the z/OS Communications Server 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 the z/OS Communications Server onward. If the response time is slow, the z/OS Communications Server or the SNA 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 TERMINAL definition.

Observing response time

The basic criterion of performance in a production system is response time. Good performance 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.

In straightforward data-entry systems, good response time implies sub-millisecond response time. In normal production systems, good response time is measured in the five to ten millisecond range. In scientific, compute-bound systems or in print systems, good response time can be one or two minutes.

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 milliseconds may be considered to be quite good. If an application requires only one access, however, a response time of three to six milliseconds 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.

Typically, the response time in the system varies with an increasing transaction rate, is gradual at first, then quickly deteriorates. The typical curve shows a sharp change when, suddenly, the response time increases dramatically for a relatively small increase in the transaction rate.

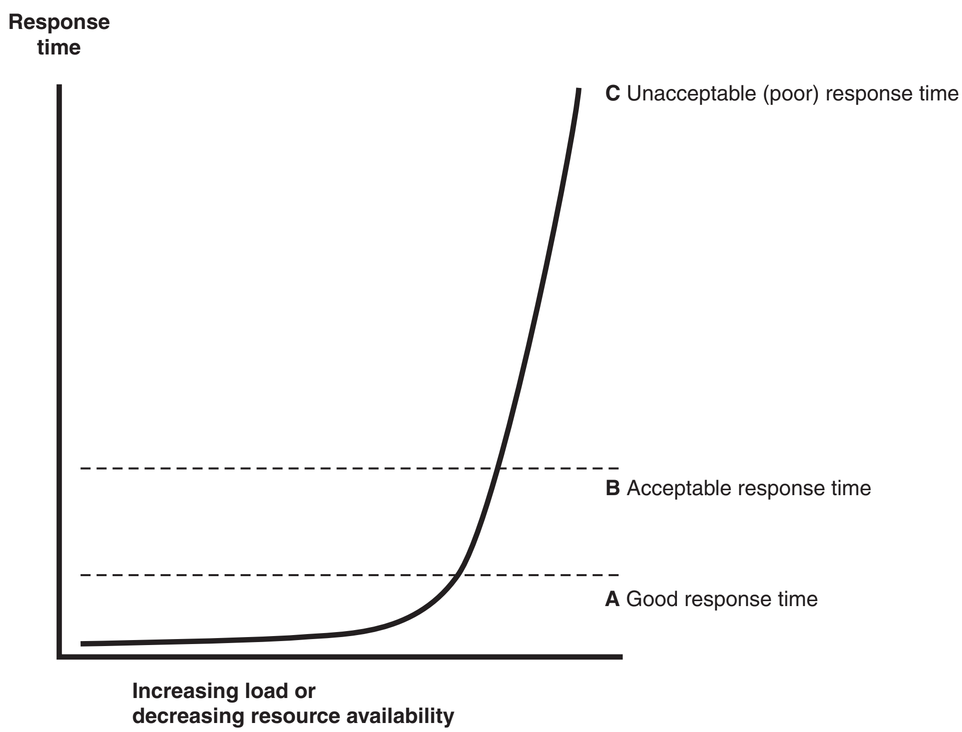


Figure 15. Graph to show the effect of response time against increasing load

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 if it is part of the processor, 80% usage might 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.

The information in this topic has been based on the assumption that CICS is the only major program running in the 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.

Poor response time: Causes and solutions

This table 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.

The first step is to check the causes by following the advice given in "Assessing the performance of your system" on page 14. When you have identified the precise causes, you can find information in Part 2, "Improving the performance of a CICS system," on page 67 on how to implement an appropriate solution.

Table 3. CICS response time checklist

Major cause	Solution
Level 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, and check for application errors and resource contention
Level 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

Table 3. CICS response time checklist (continued)

Major cause	Solution
CICS limit values exceeded	Change operands, or provide more resources, or check if errors in application
Level 3: Poor response for certain transactions only	
Identify common characteristics listed under Level 2	The solutions are as for Level 2
Lines or terminal usage	Increase capacity, or reduce data transfer, or change transaction logic
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 used by transactions	Redesign or tune application subprograms
Same access method or CICS features used by transactions	Reallocate resource or change application, and reevaluate use of feature in question
Limit conditions	Reallocate resource or change application
Level 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 terminal definitions	Redefine terminal definitions

Reducing storage stress

Storage stress occurs when there is a shortage of free space in one of the dynamic storage areas.

Storage stress can be a symptom of the following situations:

- Other resource constraints that cause CICS tasks to occupy storage for longer than usual
- A sudden large number of tasks that overwhelm available free storage
- Badly designed applications that require unreasonably large amounts of storage

CICS handles storage stress as follows:

- With decreasing free storage availability, nonresident, not-in-use programs might be deleted progressively, as CICS determines appropriate, on a least-recently-used basis. Dispatch 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 might be 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. In this way, other user tasks can 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 maximum task specification (MXT) and transaction class limits. 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 after 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 taking a number of actions. 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, DFHSM0133 or DFHSMSM0606.

Removing unwanted data set name blocks

The extended CICS dynamic storage area (ECDSA) is also used for data set name (DSN) blocks. One DSN block is created for every data set that CICS file control opens, and they are recovered at a warm or emergency restart. If an application creates a large number of temporary data sets, all with a unique name, the number of DSN blocks can increase to such an extent that they can cause a short-on-storage condition.

If application programs use temporary data sets, with a different name for every data set created, it is important that these programs remove the temporary data sets after use. See SET DSNAME in Reference > System programming for information about how you can use this command to remove unwanted temporary data sets from your CICS regions.

Language Environment® runtime options for AMODE(24) programs

Two of the default Language Environment runtime options for CICS are ALL31(ON) and STACK(ANY). This means all programs that require Language Environment must be capable of addressing 31-bit storage, that is, must be AMODE(31) when Language Environment is enabled. For AMODE(24) programs to run in a Language Environment environment, you can specify ALL31(OFF) and STACK(BELOW). However, if you change these options globally so that all programs can use them, a lot of storage will be put below the 16 MB line, which might cause a short-on-storage condition.

For more information, see "Short-on-storage conditions in dynamic storage areas" on page 105.

Purging tasks

If a CICS task is suspended for longer than its DTIMOUT value, it might be purged if SPURGE=YES is specified on the RDO transaction definition. 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.

If purging tasks is not possible or does not solve the problem, CICS stops processing. You must then cancel and restart the CICS region.

Reducing DASD paging activity

A large amount of DASD paging activity can slow down the rate at which transactions pass through the system.

About paging

The virtual storage of a processor might far exceed the size of the central storage available in the configuration. Any excess must be maintained in auxiliary storage (DASD). 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 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.

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 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 on the basis that a smaller number of concurrent transactions requires less real storage, causes less paging, and may 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 that only affects those 4K byte pages that have been touched. More real storage is required in ELSQA, however. For more information on transaction isolation and real storage see “Allocation of real storage when using transaction isolation” on page 142.

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 that 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 storage is exploited by keeping more data and programs in memory.

Program loading and paging

CICS employs MVS load under an MVS subtask to load programs. This 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.

A page-in operation causes the MVS task that 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, 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.

Reducing resource contention

Stress conditions are an indication that certain limit conditions have been reached and additional processing is required. The transactions involved must wait until resources are released.

The main limit conditions or constraints that can occur in a CICS system include those listed in Chapter 3, “Identifying CICS performance constraints,” on page 51.

To summarize, limit conditions can be indicated by the following:

- Virtual storage conditions (short-on-storage or 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 z/OS Communications Server receive-any RPLs in use (shown in the z/OS Communications Server 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.

Resolving resource problems

This table provides information about the symptoms that indicate resource problems, their causes, and their solutions.

Follow this general procedure for resolving resource problems:

1. Confirm that your diagnosis of the type of constraint is correct, by means of detailed performance analysis. “Methods of performance analysis” on page 15 describes various techniques.
2. Read “Tuning your system” on page 22 for general advice on performance tuning.
3. See the relevant sections in Part 2, “Improving the performance of a CICS system,” on page 67 for detailed information on applying the various solutions.
4. Improve virtual storage exploitation by ensuring the following:
   - Large data buffers above the 16 MB line or in Hiperspace™
   - Programs that run above the 16 MB line
   - Large amounts of real 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.

Typical resource problems, their symptoms, and their solutions:

Problem	Symptom	Solution
Excessive DASD I/O operations: the amount of I/O operations needed to locate and fetch modules from DASD storage is excessive.	• 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	• Reduce the number of I/O operations. • Tune the remaining I/O operations. • Balance the I/O operations load.
Slow transaction response on network: the average transaction response time for the network is unacceptably slow.	• 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	• Reduce the line utilization. • Reduce delays in data transmission. • Alter the network.
Slow response from remote system: the response time from a connected remote system is excessive.	• SOS condition or MXT occurs when there is a problem with a connected region • CICS takes time to recover when the problem is fixed	• 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.
Excessive use of virtual storage: excessive use of common storage is occurring, or storage is not being freed at the end of a job or address space.	• Slow response times • Multiple loads of the same program • Increased I/O operations against program libraries • High paging rates • Frequent SOS condition	• Tune the MVS system to obtain more virtual storage for CICS (increase the region size). • Make more efficient use of the dynamic storage area.
Insufficient real storage: a program issued a request for real (processor) storage that specified a variable length with a maximum value that was too high.	• High paging rates • Slow response times • MXT limit frequently reached • SOS condition often occurs	• Reduce the demands on real storage • Tune the MVS system to obtain more real storage for CICS

Problem	Symptom	Solution
Excessive processor cycling: storage buffering or cycle stealing with integrated channels is occurring, or the amount of the queue searching is excessive.	• Slow response times • Low priority transactions respond very slowly • Low priority work very slow to complete	• 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, or by tracing less frequently. • Use a faster processor.

For more information about resolving performance problems see the *Resource Measurement Facility Performance Management Guide* (SC33-7992).

Reducing storage violations

Storage violations can be reduced if CICS has storage protection and transaction isolation enabled.

CICS can detect storage violations when the duplicate storage accounting area (SAA) or the initial SAA of a TIOA storage element has become corrupted, or when 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 the following situations:
- When CICS detects an error during its normal processing of a FREEMAIN request for a TIOA storage element, and finds that the two storage check zones of the duplicate SAA and the initial SAA are not identical.
- 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.

For more information about storage violations, see the *CICS Problem Determination Guide*.

Chapter 4. Performance management and capacity planning

Performance management means monitoring and allocating existing data processing resources to applications according to a Service Level Agreement (SLA) or informal service objectives. Capacity planning is the process of planning for sufficient computer capacity in a cost-effective manner to meet the future service needs for all users.

Performance management

The goal of performance management is to make the best use of your current resources to meet your current objectives, without excessive tuning effort. To formalize your objectives, you can set up a Service Level Agreement (SLA). An SLA is a contract that objectively describes measurable performance factors, for example:
- Average transaction response time for network, I/O, processor, or total
- Transaction volumes
- System availability

A fundamental part of performance management is to measure transaction response time and break it down into components. This process shows you where tuning can be carried out for individual transactions. For effective performance management, you need to go on to measure resource requirements at the workload level. Analyzing your workload helps you to understand the behavior of your system and how workloads interact with each other.

The primary reference document for z/OS performance management is the *z/OS Resource Measurement Facility Performance Management Guide* (SC33-992-09). The following IBM publications that cover performance tuning and management are available from the z/OS information center:
- z/OS Resource Measurement Facility (RMF) Performance Management Guide
- z/OS Resource Measurement Facility (RMF) User's Guide
- z/OS MVS Initialization and Tuning Guide
- z/OS MVS Planning Workload Management

Capacity planning

Capacity planning involves asking the following questions:
- How much of your computer resources (processor, storage, I/O, network) are being used?
- Which workloads are consuming the resources (workload distribution)?
- What are the expected growth rates?
- When will the demands on current resources affect service levels?

The data that you gather, and the predictions that you make, help you to plan a schedule for upgrading your z Systems hardware, or for making additional enhancements such as adding zIIP and zAAP specialty processors to your system.

For more information about capacity planning, see the IBM Redbooks publication ABCs of z/OS System Programming, SG24-6327-01.

Related information:

z/OS V1R13.0 PDF links

Relating CICS transactions to hardware resources

Use information provided by CICS monitoring and statistics to see what hardware resources in your system are being used by CICS transactions. You can use this data for capacity planning, and also for accounting and billing purposes.

About this task

The SMF monitoring records for each CICS transaction identify the CEC machine type and CEC model number for the physical hardware environment where the CICS region is running. CEC (central electronics complex) is a commonly used synonym for CPC (central processing complex), which refers to a collection of physical hardware including main storage, one or more central processors, timers, and channels. You can use further monitoring fields to calculate the processor time that the transaction spends on a zIIP or zAAP specialty processor, and to see the processor time that the transaction could have spent on a specialty processor.

Procedure

- To link your CICS workload to a specific CPC in your system, use the information in the CECMCHTP and CECMDLID fields in the DFHTASK performance class group. CECMCHTP shows the CEC machine type for the physical hardware environment, and CECMDLID shows the CEC model number. This information is also in the monitoring domain global statistics for the CICS region, reported by the DFHSTUP statistics reporting utility and the DFH0STAT sample statistics program.

  Tip: You can use this information with the IBM Large Systems Performance Reference (LSPR) ratios to make an accurate assessment of CICS performance and relative processor capacity, particularly when considering upgrades to your z/OS hardware. For more information about the LSPR ratios, see Large Systems Performance Reference for IBM z Systems.
- To calculate the actual and potential use of zIIP or zAAP specialty processors by CICS transactions, use the information in the CPUTONCP and OFFLCPUT fields in the DFHTASK performance class group:
  - Field 436, CPUTONCP, shows the total task processor time spent on a standard processor. To calculate the task processor time spent on a specialty processor, subtract the time recorded in this field from the time recorded in field 008, USRCPUT.
  - Field 437, OFFLCPUT, shows the total task processor time that was eligible for offload to a specialty processor, but actually ran on a standard processor. To calculate the total task processor time that was not eligible for offload, subtract the time recorded in this field from the time recorded in field 436, CPUTONCP.
  - To calculate the total task processor time that was either actually spent on a specialty processor, or eligible to be spent on a specialty processor, use the following equation:
    ```
    (OFFLCPUT + (USRCPUT - CPUTONCP))
    ```

  Note: The times shown in the CPUTONCP and OFFLCPUT fields are only available when running on a system that supports the Extract CPU Time instruction service that is available on IBM System z9® or later hardware. For z/OS, Version 1 Release 13, the PTF for APAR OA38409 must also be applied.

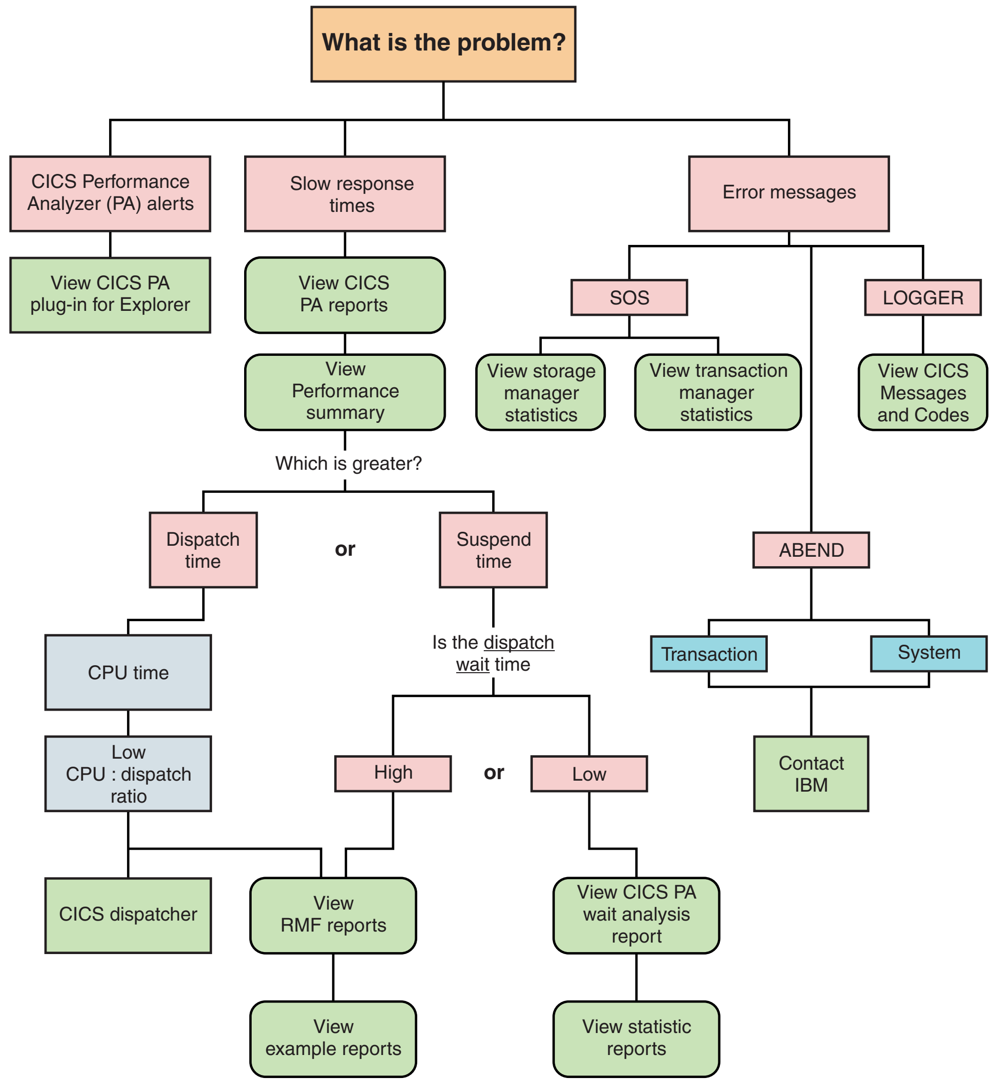
Part 2. Improving the performance of a CICS system

Tuning is a key factor in improving the performance of a CICS system. You must always tune DASD, the network, and the overall MVS system before tuning any individual CICS subsystem through CICS parameters. Before you tune a system, you must understand why your CICS system is not performing as expected.

If you have concerns about the performance of your CICS system, use the performance flow diagram as a guide to understand why your CICS system is not performing as expected, and as a guide to solve the system problems.

The red boxes are selection boxes; depending on the result of your selection you are directed to different reports. The green boxes are links which, when clicked, open topics in the information center that describe how to use the performance analysis tools to improve the performance of your CICS system.

For example, if you determine that response times in your system are too slow, view your CICS Performance Analyzer (CICS PA) reports and the performance summary. Use the performance summary and the performance analyzer report to determine whether it is the dispatch time or the suspend time of tasks that is greater. If the suspend time is greater, determine whether the dispatch wait time is low or high. If the dispatch wait time is low, click the green box on the diagram that directs you to information about reports that provide the relevant information to help you improve the performance.



To help you improve performance, you can view tuning guidelines for different aspects of CICS:

- "Using data tables" on page 197
- Chapter 6, "CICS dispatcher: performance and tuning," on page 77
- Chapter 7, "Virtual and real storage: performance and tuning," on page 85
- Chapter 8, "CICS storage protection facilities: Performance and tuning," on page 145
- Chapter 9, "Tuning with Language Environment," on page 147
- Chapter 10, "Java applications: performance and tuning," on page 151
- Chapter 11, "MVS and DASD: performance and tuning," on page 153
- Chapter 12, "Networking and the z/OS Communications Server: performance and tuning," on page 155
- Chapter 13, "CICS MRO, ISC, and IPIC: performance and tuning," on page 171
- Chapter 14, "CICS VSAM and file control: Performance and tuning," on page 183
- Chapter 15, "Database management for performance," on page 219
- Chapter 16, "CICS logging and journaling: Performance and tuning," on page 225

Chapter 5. CICS Transaction Manager: performance and tuning

The CICS Transaction Manager domain provides transaction-related services.

The services provided by the domain are used to:
- Create tasks
- Terminate tasks
- Purge tasks
- Inquire on tasks
- Manage transaction definitions
- Manage tranclass definitions

The transaction manager domain also provides a transaction environment to enable other CICS components to implement transaction-related services.

For more information about transactions, see Transaction statistics.

Setting the maximum task specification (MXT)

The **MXT** system initialization parameter limits the total number of concurrent user tasks in the CICS system.

The **MXT** parameter controls unconstrained resource demand, particularly virtual storage usage in order to avoid short-on-storage (SOS) conditions. This parameter also affects the amount of storage allocated to the kernel stack segment and controls contention for resources, the length of queues (which can avoid excessive processor usage), and real storage usage.

The value of **MXT** affects the storage use in the CICS address space. You must ensure that enough storage is available for other users in the dynamic storage areas (DSAs).

The **MXT** parameter controls the number of user tasks that are eligible for dispatch. **MXT** can be set at startup, or by using the **SET SYSTEM** command. When **MXT** is set, the kernel and dispatcher attempt to preallocate sufficient control blocks to guarantee that the number of user tasks specified by the **MXT** value can be created concurrently. Most of the storage for this preallocation is obtained from the CDSA or ECDSA, although a small amount of MVS storage is required for each task (approximately 256 bytes in 31-bit storage, above the 16 MB line, and 32 bytes in 24-bit storage, below the 16 MB line, for each user task). The **MXT** value is interrelated with the z/OS REGION size, and the DSA size limits that you set (the **DSALIM** and **EDSALIM** parameters). See “Setting the limits for CICS storage” on page 94.

The **MXT** system initialization parameter has a default value of 250, a minimum setting of 10, and a maximum setting of 2000. Initially, set **MXT** to the total number of concurrent user tasks that you require in your system.

If you set the **MXT** value too high, throughput and response time can suffer when system resources (processor, real storage, and virtual storage) are constrained or resource contention occurs, for example file strings or buffers, DB2 threads, ENQs and so on. Also, if you set the **MXT** value too high at startup, CICS forces a smaller maximum number of tasks consistent with the available virtual storage.

Conversely, if you set the **MXT** value too low, throughput and response time can also suffer due to excessive queuing delays even though system resources (processor, real storage, and virtual storage) are not constrained.

Monitor the performance of the CICS region to ensure that the response time and other time components (such as dispatch time and suspend time) for your transactions are acceptable. In some systems, setting **MXT** too high might increase resource contention to a level that causes additional delays for transactions. You can use the transaction manager global statistics from the DFH0STAT and DFHSTUP utility programs to monitor the **MXT** value.

If performance tuning for HTTP connections is enabled, when the region is at capacity, instead of queuing HTTP requests on MXT in CICS (which involves internal processing and storage requirements), requests are queued outside of CICS in the TCP/IP backlog. MXT is no longer exceeded by a surge of HTTP requests (externalized as the **XMGPQT** field in the transaction manager global statistics), but the number of times MXT is reached (externalized as the **XMGTAMXT** field in the transaction manager global statistics) might increase. It can occur if the region remains under stress when CICS processes each request from the backlog queue; as the task to process the request is attached CICS goes back to MXT. Once the levels on MXT are decreased CICS will accept the next request, and its transaction might cause CICS to go back to MXT. It does not indicate that MXT needs to be increased, it shows that its current value is correctly protecting CICS from unconstrained resource demand.

In addition, the following performance data fields in the DFHTASK group are useful to assess the relationship between the task load during the life of a transaction, and the performance of the transaction:

- CURTASKS records the current number of active user transactions in the system at the time the user task was attached.
- MAXTASKS records the current setting for the maximum number of tasks for the CICS region at the time the user task was attached.
- MXTDELAY records the elapsed time waiting for the first dispatch when the delay is because the **MXT** value is reached.

To alter the **MXT** value while CICS is running, you can use the **SET SYSTEM MAXTASKS** command. If you set the **MXT** value too high while CICS is running, the error message: "CEILING REACHED" is displayed. The CICS transaction manager statistics show the number of times the **MXT** ceiling has been reached.

Note: If the MAXOPENTCBS or MAXXPTCBS system initialization parameters have not been specified, then the **MXT** parameter also sets the MAXOPENTCBS and MAXXPTCBS parameters.

Important: Before you change the **MXT** value, review the information in Open TCB pools in Improving performance.

Using transaction classes (MAXACTIVE) to control transactions

Transaction classes give you a mechanism to limit the number of CICS tasks in 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.

Use the *MAXACTIVE* attribute of the transaction class definition (TRANCLASS) to control a specific set of tasks that are heavy resource users, tasks of lesser importance (for example, "Good morning" broadcast messages), and so on, allowing processor time or storage for other tasks. Together with the **MXT** system initialization parameter, transaction classes control the transaction mix, that is, ensuring 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 tasks, or put all user tasks into separate classes. Suggested classes are simple inquiries, complex inquiries 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 tasks that consume particularly large amounts of resource, but that do not exceed the MAXACTIVE ceiling frequently. Do not use transaction classes 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.

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). CICS defines two transaction classes for its own use, DFHTCLSX and DFHTCLQ2. For information about the effects these have, see "Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use" on page 177.

Specifying a transaction class purge threshold (PURGETHRESH)

The PURGETHRESH attribute of the transaction class definition limits the number of tasks that 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.

The tasks 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. The **MXT** system initialization parameter limits the total number of tasks that have already been admitted to the system within TRANCLASS constraints.

The length of the queue of tasks waiting to be started in a transaction class is limited by the PURGETHRESH attribute of that class. Any new transaction which would cause the limit to be reached is ended with the abend code AKCC. Tasks that were queued before the limit was reached are allowed to continue waiting until they can be executed.

The PURGETHRESH attribute should be specified only where the transaction load in a transaction class 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 transaction classes 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 stop and the associated transaction class fills (up to the value defined by MAXACTIVE) with tasks that are unable to complete their work in the AOR. New transactions 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.

The size of each entry in the queue is the size of a transaction (256 bytes), plus the size of an interval control element (ICE) secure storage extension (2108 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 purge threshold 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 PURGETHRESH 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. The PURGETHRESH attribute of a TRANCLASS is used to set the limit of the queue for that transaction class. The default action is not to limit the length of the queue.

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. These are the most useful statistics for monitoring queue lengths:

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 monitor the number of AKCC abends in the CSMT log. The AKCC 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.

Prioritizing tasks

Prioritization is a method of giving specific tasks preference in being dispatched. Priority is specified in the TERMINAL definition (TERMPRIORITY), a transaction in a TRANSACTION definition (PRIORITY), and in the priority field of the user segment of the external security manager (ESM), (OPPRTY).

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 PRTYAGE system initialization parameter also influences the dispatching order; for example, the default value **PRTYAGE**=1000 causes the task's priority to increase by 1 every 1000ms it spends on the ready queue. 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. Therefore, a low priority task might 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 the task is dispatched. PRTYAGE should usually remain at its default value, unless certain transactions get stuck behind higher priority transactions during very busy periods.

Note: Non-terminal transactions are attached with a priority value based on the transaction priority from the TXD, and the operator priority, while terminal control based tasks are attached with only the transaction priority. When the task first gets dispatched the operator priority is added in. For this reason, terminal and non-terminal based tasks must not be managed through the same transaction class, because a steady stream of non-terminal based transactions could take precedence over other terminal control based transactions on a sufficiently busy system.

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, you can improve overall throughput and response time. Prioritization can minimize resource usage of certain resource-bound transactions. 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 in a system. CICS prioritization is not interrupt-driven as is the case with operating system prioritization, but 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.

You should 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 in 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 and 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. Here are some examples:

- 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 use transient data intrapartition queues that have a destination of terminal defined and a trigger level that is greater than zero).
  - Batch update controlling tasks.

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.

Chapter 6. CICS dispatcher: performance and tuning

You can tune the performance of the CICS dispatcher by specifying dispatch intervals. You specify dispatch intervals by setting system initialization parameters for interval control values and other parameters such as FORCEQR, MROBTCH, PRTYAGE, and SUBTSKS.

For more information about dispatcher statistics, see “Dispatcher domain statistics” on page 468.

Open TCB management

The open transaction environment (OTE) is an environment where CICS application code can use non-CICS services (facilities outside the scope of the CICS API) inside the CICS address space, without causing wait issues on the quasi-reentrant task control block (QR TCB).

Applications that exploit the open transaction environment run on their own open TCB, rather than on the QR TCB. CICS does not perform subdispatching on an open TCB, whereas it does on the QR TCB. If the application that is running on an open TCB calls a non-CICS service that blocks the TCB, the TCB blocking does not affect other CICS tasks. For more information about writing applications to exploit the open transaction environment, see Multithreading: Reentrant, quasi-reentrant, and threadsafe programs in Developing applications.

TCB modes

Each open TCB mode has a 2-character identifier to indicate its specific purpose, and is handled by CICS in a different way.

L8 mode TCBs and L9 mode TCBs

These TCBs are used as follows:

- L8 TCBs are used for CICS-key application programs that are defined as API(OPENAPI) by their PROGRAM resource definition.
- L8 TCBs are used for CICS-key application programs that are defined as CONCURRENCY(REQUIRED), API(CICSAPI) by their PROGRAM resource definition.
- L8 TCBs are used when programs need access to a resource manager through a task-related user exit (TRUE) that was enabled using the OPENAPI option on the ENABLE PROGRAM command. Task-related user exits always run in CICS key.
- L8 TCBs are used by CICS when accessing document templates and HTTP static responses that are stored in z/OS UNIX System Services files.
- L8 TCBs are used for web service requests and parsing XML CICS programs that run in CICS key and are defined as OPENAPI.
- L9 TCBs are used for application programs that run in user key and are defined as OPENAPI.

CICS operates with an OPENAPI task-related user exit, and therefore uses L8 TCBs, when it is connected to the following products:

- WebSphere MQ, using the CICS WebSphereMQ adapter
- DB2, using the CICS DB2 Attachment Facility.
- IMS Version 12 or later, using the CICS DBCTL Database Adapter Transformer (DFHDBAT).

Other IBM products, for example IP CICS Sockets and the z/OS Integrated Cryptographic Service Facility (ICSF), can also use an OPENAPI enabled task-related user exit. For more information about managing IP CICS sockets, see z/OS Communications Server: IP CICS Sockets Guide. For more information about the CICS-ICSF Attachment Facility, see z/OS Cryptographic Services ICSF System Programmer's Guide.

SP mode TCB and S8 mode TCBs

These TCBs are used by CICS to manage SSL connections and requests to LDAP using the DFHDDAPX XPI interface. The S8 TCBs run in a single enclave, which is owned by the SP TCB and also contains the SSL cache. An S8 TCB is allocated to a task from the SSL pool, but is locked only for the period of time that it takes to perform functions such as an SSL handshake or an LDAP request. After this function is complete, the TCB is released back into the SSL pool to be reused.

In UNIX System Services (USS), the **MAXTHREADS** and **MAXTHREADTASKS** parameters can be used to restrict the number of pthreads that a USS process can own. Each SSL TCB requires a pthread and an MVS task. You must therefore ensure that the values of these USS parameters exceed the value of the **MAXSSLTCBS** system initialization parameter. If you do not set a large enough value for **MAXTHREADS** or **MAXTHREADTASKS** and CICS reaches one of these limits while attempting to attach an SSL TCB, CICS issues error message DFHDS0002 severe error code X'0137' from DFHDSIT.

TP mode TCB and T8 mode TCBs

These TCBs are used by a JVM server to process requests for Java™ programs. The JVM server is a runtime environment that can handle multiple concurrent requests for Java applications in a single JVM. The TP mode TCB owns the Language Environment enclave and the pool of T8 TCBs. Each JVM server that is running in the CICS region has one TP TCB and at least one, but not more than 256, T8 TCBs. A T8 TCB is allocated to a task from the THRD pool of the appropriate JVM server, but is locked only for the period of time that it takes to perform the system processing. T8 TCBs are not shared between JVM servers.

Each T8 TCB requires a pthread and an MVS task. The maximum number of T8 TCBs that is allowed for the CICS region is 2000. In z/OS UNIX, you can use the **MAXTHREADS** and **MAXTHREADTASKS** parameters to restrict the number of pthreads that a z/OS UNIX process can own. You must therefore ensure that the values of these parameters exceed the maximum number of T8 TCBs. If you do not set a large enough value for **MAXTHREADS** or **MAXTHREADTASKS** and CICS reaches one of these limits while attempting to attach a T8 TCB, CICS issues error message DFHDS0002 severe error code X'0137' from DFHDSIT. For more information about the thread limits of JVM servers, see .

X8 mode TCBs and X9 mode TCBs

Both these TCBs are used to run C and C++ programs compiled with the XPLINK option. X8 TCBs are used for programs in CICS key, and X9 TCBs are used for programs in user key. Each instance of an XPLink program uses one X8 or X9 TCB. For more information about using XPLink, see the *CICS Application Programming Guide*.

Open TCB pools

CICS manages open TCBs in *pools*. A pool contains open TCBs that are used for the same purposes. A pool can consist of some TCBs that are allocated to tasks, and others that have been freed by applications and are available for reuse.

CICS can create or attach open TCBs in each pool up to the limit set for the pool. The maximum number of TCBs allowed in each pool is set as follows:

- The **MAXOPENTCBS** system initialization parameter, if specified, sets the value for the open TCB pool. If the MAXOPENTCBS system initialization is not specified, CICS sets the limit for the L8 and L9 mode open TCB pool automatically based on the maximum number of tasks specified for the CICS region (the MXT value), using the following formula: `(2 * MXT Value) + 32`.
- The **MAXSSLTCBS** system initialization parameter specifies the value for the SSL TCB pool.
- The **MAXTHRDTCBS** system initialization parameter specifies the value for the JVM server THRD TCB pool. The number of threads reserved for each JVM server is the THREADLIMIT value on the JVMSERVER resource, plus 1, up to a limit of 2000.
- The **MAXXPTCBS** system initialization parameter, if specified, sets the value for the XP TCB pool. If the **MAXXPTCBS** system initialization is not specified, CICS sets the limit for the X8 and X9 mode XP TCB pool automatically to a value equal to the maximum number of tasks specified for the CICS region (the MXT value)

When an application makes a request that requires an open TCB, CICS first tries to find a suitable TCB that is available for reuse in the appropriate pool. CICS can match a request with an available TCB of the correct mode only if the TCB is for the correct subspace. CICS attaches a new TCB when it cannot find a suitable match with a free TCB for the correct subspace, provided that the limit for the pool has not been reached.

A CICS task is allowed as many X8 and X9 TCBs as it requires, and these TCBs are kept only until the program finishes. However, each CICS task is allowed at most one L8 and one L9 TCB, and it keeps an L8 and an L9 TCB from the time it is allocated to the end of the task, reusing it for further requests as needed. The TCBs then become free, and CICS can allocate them to another task or destroy them.

Sometimes CICS cannot find a suitable match for an application's request, and the limit for the pool has been reached. In this situation, CICS might fulfil the request by destroying a free TCB in the pool that has the wrong subspace or mode, and replacing it with a TCB with the right mode and subspace. This technique is called *stealing*. Stealing can be costly on performance, depending on the type of open TCB, so CICS avoids it where it makes sense to do so. CICS maintains statistics of excess TCB management and TCB stealing activities in the CICS dispatcher TCB mode and TCB pool statistics.

If the number of TCBs is at the limit for the pool and there is no free TCB to steal, the task is suspended with an OPENPOOL wait until a TCB becomes free, or the limit for the pool is increased.

To minimize the impact on storage, CICS attempts to balance the number of open TCBs in each pool against current needs. If CICS finds free TCBs in a pool, it gradually reduces the excess number by detaching them, freeing the resources used by the excess TCBs.

MAXSSLTCBS

You can use the dispatcher TCB statistics from the DFH0STAT and DFHSTUP utility programs to monitor the S8 TCBs in the SSL pool. The maximum number of TCBs is set by the **MAXSSLTCBS** system initialization parameter.

If you want to improve the performance of SSL, you can use the dispatcher reports to find out if there are many tasks waiting for an S8 TCB. Also look at the number of tasks that have queued. If both fields report a large number, increase the maximum number of S8 TCBs. If you have few tasks queued, but many waits, you can decide whether you want to increase the number of S8 TCBs. Increasing the number by one or two could make a difference to the number of waits and reduce the tasks queued, without causing significant overheads in storage.

The maximum number of S8 TCBs that you can set is 1024. However, setting many S8 TCBs can also affect performance because of the amount of storage used. If CICS runs out of storage, you get a TCB attach failure. This failure is reported in the dispatcher reports for the S8 TCB mode statistics.

For more information about the MAXSSLTCBS system initialization parameter, see MAXSSLTCBS system initialization parameter in Reference -> System definition.

Interval control value parameters: ICV, ICVR, and ICVTSD

The interval control values (ICVs) are specified in the system initialization table (SIT) to set a new value or by overrides. Setting the ICV parameters correctly can help to improve performance by reducing processor usage for low utilization CICS regions.

CICS has three types of interval control value parameters:

Interval control value (ICV)
: 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. This time interval can be any integer in the range 100 through 3600000 milliseconds (specifying an interval up to 60 minutes). A typical range of operation might be 100 through 2000 milliseconds. For more information, see ICV system initialization parameter in Reference -> System definition.

Interval control value for runway tasks (ICVR)
: The **ICVR** system initialization parameter specifies the default runaway task time interval in milliseconds as a decimal number. You can specify zero, or a number in the range 500 through 2700000, in multiples of 500. CICS rounds down values that are not multiples of 500. This is the RUNAWAY interval that is used by transactions defined with RUNAWAY=SYSTEM. For more information, see ICVR system initialization parameter in Reference -> System definition.

Interval control value for terminal scan delay (ICVTSD)
: The **ICVTSD** system initialization parameter was used in earlier releases to limit how quickly CICS dealt with some types of terminal output requests made by applications, in order to spread the overhead for dealing with the requests. The range is 0 through 5000 milliseconds. Specifying a nonzero value was sometimes appropriate where the CICS system used non-SNA networks. However, with SNA and IPIC networks, setting ICVTSD to 0, which is the default, is appropriate to provide a better response time and best virtual storage usage.

MROBTCH

The **MROBTCH** system initialization parameter specifies how many events in a region can be accumulated in a batch before posting.

The region is started so that it can process the requests. The batching of multiregion operation (MRO) requests includes some non-MRO events:
- VSAM physical I/O completion events
- Subtasked request completion (mostly VSAM)
- DL/I request completion implemented through DBCTL

The value of the **MROBTCH** parameter can be in the range of 1 through 255, and the default is 1. Using this batching mechanism, you can spread the dispatch resource usage in CICS over several tasks. If the value is greater than 1 and CICS is in a system wait, CICS is not posted for dispatch until the specified number of events has occurred. Events include MRO requests from connected systems or DASD I/O and CHANGE_MODE processing. For these events, CICS is dispatched after:
- The current batch fills up (the number of events equals **MROBTCH**)
- An ICV interval expires

The time interval that you specify in the ICV parameter should be low enough to prevent undue delay to the system.

During periods of low utilization, a value of **MROBTCH** greater than 1 can result in increased transaction response times. Transactions that issue I/O requests might be delayed because of an increased FCIOWAIT value. For more information about the effect of MROBTCH on performance, see “Batching requests (MROBTCH)” on page 178.

FORCEQR

The **FORCEQR** system initialization parameter specifies whether you want CICS to force all CICS API user application programs that are specified as threadsafe to run under the CICS quasi-reentrant (QR) task control block (TCB), as if they were specified as quasi-reentrant programs.

If your programs are defined as quasi-reentrant, CICS always calls them under the CICS QR TCB. The requirements for a quasi-reentrant program in a multithreading context are less stringent than if the program were to execute concurrently on multiple TCBs. CICS requires that an application program is reentrant so that it guarantees consistent conditions. In practice, an application program may not be truly reentrant; CICS expects "quasi-reentrancy". This means that the application program should be in a consistent state when control is passed to it, both on entry, and before and after each **EXEC** CICS command. Such quasi-reentrancy guarantees that each invocation of an application program is unaffected by previous runs, or by concurrent multi-threading through the program by multiple CICS tasks.

CICS quasi-reentrant user programs (application programs, user-replaceable modules, global user exits, and task-related user exits) are given control by the CICS dispatcher under the QR TCB. When running under this TCB, a program can be sure that no other quasi-reentrant program can run until it relinquishes control during a CICS request, at which point the user task is suspended, leaving the program still "in use". The same program can then be reinvoked for another task, which means the application program can be in use concurrently by more than one task, although only one task at a time can actually be executing.

Running application with programs defined as threadsafe to use OTE, such as CICS DB2 applications, could cause problems if one or more programs is not threadsafe. Using the **FORCEQR** system initialization parameter, you can force all your applications onto the QR TCB.

Forcing applications on the QR TCB is useful in production regions where you cannot afford to have applications out of service while you investigate the problem.

The default for this parameter is **FORCEQR=NO**, which means that CICS honors the **CONCURRENCY** attribute on your program resource definitions. As a temporary measure, while you investigate and resolve problems connected with threadsafe-defined programs, you can set **FORCEQR=YES**. When all problems have been resolved, resetting **FORCEQR=NO** makes all programs resume use of open TCBs under the OTE.

The **FORCEQR** parameter applies to all application programs that are restricted to the current CICS programming interfaces (programs that specify API(CICSAPI)). The parameter does not apply to any of the following programs:
- Java programs that are run in JVM
- C or C++ programs using XPLINK
- OPENAPI programs
- Programs defined with CONCURRENCY(REQUIRED)

The **FORCEQR** parameter applies to all programs defined as threadsafe that are not used as task-related user exits, global user exits, or user-replaceable modules.

PRTYAGE

The system initialization parameter, **PRTYAGE**, can be set to a value that determines the rate at which tasks in the system have their priorities aged.

The priority of a task within CICS determines the order it is dispatched. Tasks can have priority values of 1 through 255. If a task's first dispatch is too slow, changing the priority to a higher value shortens the dispatch time. You have no control over the priorities of CICS system tasks. Adjusting the value of **PRTYAGE** does not control the priorities of tasks, only how CICS sets the priorities of tasks. Altering the value of **PRTYAGE** affects the rate at which tasks are dispatched.

SUBTSKS

The **SUBTSKS** system initialization parameter specifies the number of task control blocks (TCBs) that CICS uses to run tasks in concurrent mode.

The value of the **SUBTSKS** parameter is either 1 or 0. The value of 1 turns subtasking on and the value of 0 turns subtasking off.

Using the value of 0, which is the default value for **SUBTSKS**, CICS runs under the quasi-reentrant (QR) TCB and runs all applications under the QR TCB. At this value, CICS also runs tasks that open and close files under the resource-owning mode TCB.

If the parameter value is set to 1, CICS runs under the resource-owning TCB and the QR TCB, and uses an additional TCB, a concurrent mode TCB, to perform system subtasking.

Interpreting dispatcher statistics

Use TCB dispatcher statistics and dispatcher TCB pool statistics to understand how the CICS dispatcher is performing.

For more information about dispatcher statistics, see "Dispatcher domain statistics" on page 468.

TCB statistics

The task control block (TCB) dispatcher statistics report the amount of CPU time consumed by each CICS TCB since the last time statistics were reset.

To calculate the approximate time since CICS statistics were last reset, add the values of "Accum time in MVS wait" and "Accum time dispatched". To calculate the percentage usage of each CICS TCB, divide the value of "Accum CPU time/TCB" by the time since CICS statistics were last reset (as calculated earlier).

The "Accum CPU time/TCB" value does not include uncaptured time. Therefore, when you use this calculation, even a very busy CICS TCB is noticeably less than 100% busy. If the calculation indicates that a CICS region is more than 70% busy, you are approaching the capacity of the region. However, the 70% calculation can be only approximate. The capacity of the region depends on such factors as the workload in operation, the mix of activity in the workload, and which release of CICS you are currently using. You can use Resource Measurement Facility (RMF) to obtain a definitive measurement to use in your calculation, or you can use RMF with your monitoring system. For more information, see the z/OS Resource Measurement Facility (RMF) Performance Management Guide.

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

TCB modes are as follows:

QR There is always one quasi-reentrant mode TCB. It is used to run quasi-reentrant CICS code and non-threadsafe application code.

FO There is always one file-owning TCB. It is used for opening and closing user data sets.

RO There is always one resource-owning TCB. The RO TCB is used for loading programs, unless the command to load the program (EXEC CICS LOAD, XCTL, or LINK) is issued by an application that is currently running on an open TCB. In that situation, the open TCB is used to load the program instead of the RO TCB. The RO TCB is also used for opening and closing CICS data sets, issuing RACF® calls, and similar tasks.

The CICS loader domain global statistics record the number of program load operations that took place on the RO TCB, and the time taken for them. You can compare these values to the overall statistics for the number and time of program load operations, to see the proportion of program load operations that took place on open TCBs instead of the RO TCB.

CO The optional concurrent mode TCB is used for processes that can safely run in parallel with other CICS activity such as VSAM requests. Define the system initialization parameter **SUBTSKS** using the value 0 or 1 to specify whether there is a CO TCB.

D2
The D2 mode TCB is used to stop DB2 protected threads. Protected threads are stopped in the normal purge cycle, or when a user issues the **DSNC DISCONNECT** *plan-name* command, which stops the protected threads for a plan immediately.

SZ
The single optional SZ mode TCB is used by the FEPI interface.

RP
The single optional RP mode TCB is used to make ONC/RPC calls.

EP
The EP mode TCBs are used to run event processing in a CICS region. The TCBs either dispatch events to an appropriate EP adapter or defer filtering of system events.

L8
A task has an L8 mode TCB for its sole use when it calls a program that is enabled with the OPENAPI option and is defined with EXECKEY=CICS, or when it calls a task-related user exit program that is enabled with the OPENAPI option. An L8 TCB is used when CICS uses the CICS-MQ adapter to connect to WebSphere MQ Version 6 or later and when CICS connects to DB2 Version 8 or earlier.

L9
A task has an L9 mode TCB for its sole use when it calls a program that is enabled with the OPENAPI option and is defined with **EXECKEY=USER**.

SO
The SO mode TCB is used to make calls to the socket interface of TCP/IP.

SL
The SL mode TCB is used to wait for activity on a set of listening sockets.

S8
A task uses an S8 TCB if it needs to use the system Secure Sockets Layer (SSL). A task also uses an S8 TCB if it needs to use LDAP over the DFHDDAPX XPI interface. The TCB is used only for the duration of the SSL negotiation or the LDAP request. On completion, the TCB is released back into the SSL pool to be reused.

SP
The SP mode TCB is used for socket pthread owning tasks. It manages the SSL pool of S8 TCBs and owns the Language Environment enclave that contains the SSL cache.

T8
A Java application running in a JVMSERVER uses a T8 TCB. The T8 will also be used for DB2 requests from Java applications when CICS is connected to DB2 Version 9 or later.

TP
The TP mode TCB owns and manages the Language Environment enclave, the JVM, the THRD TCB pool, and T8 TCBs of a JVM server.

X8
A task has an X8 mode TCB for its sole use when it calls a C or C++ program that is compiled with the XPLINK compiler option and defined with **EXECKEY=CICS**. The CICS-DB2 task-related user exit may use a X8 TCB if it is run with CONCURRENCY(REQUIRED) and API(CICSAPI).

X9
A task has an X9 mode TCB for its sole use when it calls a C or C++ program that is compiled with the XPLINK compiler option and defined with **EXECKEY=USER**.

Chapter 7. Virtual and real storage: performance and tuning

Learn about virtual and real storage use in a z/OS system and in CICS regions, and start to monitor performance and tune your use of storage.

Procedure

1. Understand how virtual and real storage is arranged and managed in a z/OS address space. For information about address spaces in z/OS, read the following z/OS documentation:
   - For information about 24-bit and 31-bit storage (below the bar) in an address space, see Virtual Storage Overview in the z/OS MVS Initialization and Tuning Guide.
   - For information about 64-bit (above-the-bar) storage in an address space, see Using the 64-bit Address Space in the z/OS MVS Programming: Extended Addressability Guide.
2. Understand how virtual storage is arranged and managed when a z/OS address space is used by a CICS region. For information about how CICS uses a z/OS address space, see "CICS virtual storage."
3. Monitor and measure the use of virtual storage across your z/OS system using z/OS performance and monitoring tools, such as the z/OS Resource Measurement Facility (RMF). For an overview of RMF, see "Resource measurement facility (RMF)" on page 34. For further information, see the z/OS Resource Measurement Facility (RMF) User's Guide.
4. Monitor and measure the use of virtual storage and the size of the dynamic storage areas (DSAs) in each of your CICS regions, using the following facilities:
   - The CICS storage manager statistics
   - The reports produced by the sample statistics program DFH0STAT
   - CICS formatted dumps for the loader domain and storage domain
5. Tune the use of storage across your z/OS system. If you are using RMF, for explanations of the RMF reports relating to CICS and to storage, and strategies for tuning, see the z/OS Resource Measurement Facility (RMF) Performance Management Guide and z/OS Resource Measurement Facility (RMF) Report Analysis.
6. Tune the use of storage in each of your CICS regions, using the methods and suggestions in this section of the information. Concentrate on the areas of storage that seem to be the most different from your expectations.

CICS virtual storage

Each CICS region operates in its own z/OS address space. The storage available in a z/OS address space is divided into several different areas.

Figure 16 on page 86 shows an outline of the storage available in a z/OS address space. Although the theoretical upper limit for this virtual storage is extremely high, there are practical limits to real storage. For this reason, in z/OS, each address space is subject to **REGION** and **MEMLIMIT** parameters that limit the amount of storage the address space can use.

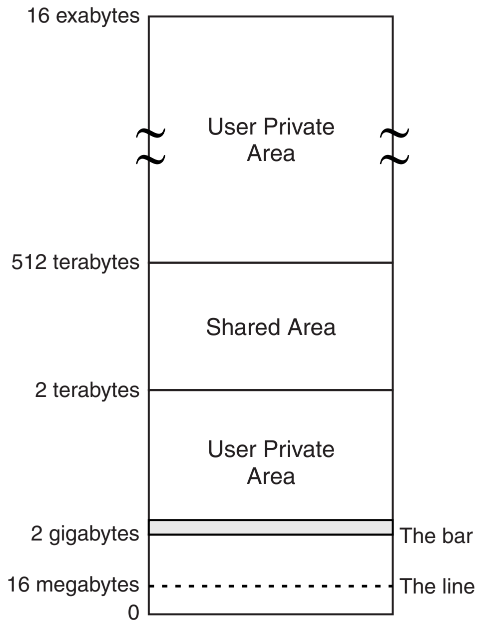


Figure 16. z/OS Address Space

CICS uses and manages virtual storage in three areas of its z/OS address space:

Storage below the line (0 MB to 16 MB)

The storage in this area is 24-bit storage.

Addresses below the 16 MB address are accessed by 24-bit addressing, and programs can use this storage when they run in AMODE 24 or higher. The 16 MB address is known as the line, so 24-bit storage is also called storage *below the line*.

Storage above the line (16 MB to 2 GB)

The storage in this area is 31-bit storage.

Addresses above the 16 MB address but below the 2 GB address are accessed by 31-bit addressing, and programs can use this storage when they run in AMODE 31 or higher. The 16 MB address is known as the line, so 31-bit storage is also called storage *above the line*.

The area that separates the virtual storage area below the 2 GB address from the user private area is known as *the bar*. 24-bit and 31-bit storage are in storage below 2 GB and can together be referred to as storage *below the bar*.

Storage above the bar (4 GB to a theoretical 16 exabytes)

The storage in this area is 64-bit storage.

The area that separates the virtual storage area below the 2 GB address from the user private area is known as the bar, and 64-bit storage is also known as storage *above the bar*.

The storage above the bar comprises a user private area between 4 GB and 2 terabytes, a shared area between 2 terabytes and 512 terabytes, and a user private area between the end of the shared area and 16 exabytes.

Addresses above the bar are accessed by 64-bit addressing, and programs can use this storage when they run in AMODE 64.

In each private area of storage, virtual storage is used for the following purposes:

CICS dynamic storage areas
The dynamic storage areas are used to supply the storage requirements of CICS, access methods, and applications running in CICS. See "CICS dynamic storage areas" on page 88.

MVS storage
MVS storage is available to the operating system to perform region-related services. See "64-bit MVS storage" on page 129 and "MVS storage below 2 GB" on page 130.

Note: This information and the following topics refer to other products installed with CICS, and is valid at the time of writing. For other products installed with CICS, always check the information for the versions of those products that you are using.

CICS region size

The amount of virtual storage for the address space in which CICS runs is specified by the z/OS **REGION** and **MEMLIMIT** parameters.

- The z/OS **REGION** parameter specifies your request for an amount of 24-bit and 31-bit storage, that is, storage below the bar. Up to 2047 MB of storage can be requested, but you must leave enough storage for the region-related services that require MVS storage below 2 GB.
- The z/OS **MEMLIMIT** parameter specifies the limit of 64-bit (above-the-bar) storage for the CICS region. A CICS region needs a **MEMLIMIT** value of at least 6 GB. The default value in z/OS for **MEMLIMIT** is 2 GB.

Reassess your settings for the **REGION** and **MEMLIMIT** parameters when you upgrade to a new release of CICS. Also reassess your settings when you install a new release of z/OS or a non-CICS subsystem. Changes in CICS can alter the requirements for 24-bit, 31-bit, and 64-bit storage for the CICS DSAs. Changes to other products can alter the requirements for MVS storage outside the CICS DSAs; for example, the requirements for 24-bit storage might reduce.

You cannot alter the **REGION** or **MEMLIMIT** values for the CICS region while CICS is running. You can specify new values on the next start of the CICS region. For instructions, see Setting address space storage limits for a CICS region in the *CICS System Definition Guide*.

You can specify the **REGION** parameter in different ways to request a specific amount of storage, or to request all the available 24-bit or 31-bit private storage. The resulting region size below the bar can be unpredictable. The z/OS message IEF374I reports the total amount of storage below the bar that z/OS assigns to a CICS region. The VIRT=nnnK portion of the message shows the 24-bit storage, and the EXT=nnnK portion shows the 31-bit storage. The CICS sample statistics program, DFH0STAT, produces a report that contains this information. You can also use RMF to monitor your use of storage in more detail.

If you plan to increase in the **REGION** value, remember the following points:

- Be aware of storage requirements in the high private area in 24-bit and 31-bit storage. Some of this storage is used by the z/OS Communications Server and other programs. An increase in the 24-bit or 31-bit storage allocated to CICS decreases the storage available for the items in the high private area. These items are the local system queue area (LSQA), scheduler work area (SWA), and

subpools 229 and 230. A shortage in these subpools can cause S80A, S40D, and S822 abends. For more information about the high private area and LSQAs, see "High private area" on page 134.
- If you increase the **REGION** value, remember to increase your values for the CICS system initialization parameters **DSALIM** and **EDSALIM** as appropriate, otherwise CICS cannot use the additional storage.

For more information about the **REGION** and **MEMLIMIT** parameters and how they apply to z/OS address spaces, see the following z/OS information:
- For information about 24-bit and 31-bit storage (storage below the bar) in an address space, see Virtual Storage Overview in the z/OS MVS Initialization and Tuning Guide.
- For information about 64-bit (above-the-bar) storage in an address space, see Using the 64-bit Address Space in the z/OS MVS Programming: Extended Addressability Guide.
- REGION Parameter in the z/OS MVS JCL Reference.
- MEMLIMIT Parameter in the z/OS MVS JCL Reference.

If the total amount of virtual storage required for your CICS regions increases, you might need to review the amount of space allocated for supervisor call (SVC) dumps that are requested by CICS, and the amount of auxiliary storage available. For information about SVC dump data set management, see z/OS MVS Diagnosis Tools and Service Aids. For information about auxiliary storage management, see the z/OS MVS Initialization and Tuning Guide.

CICS dynamic storage areas

The dynamic storage areas (DSAs) supply CICS tasks with storage to run transactions and are essential for CICS operation. The DSAs in 24-bit storage are the CDSA, UDSA, SDSA, and RDSA. The DSAs in 31-bit storage are the ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA. The DSAs in 64-bit storage are the GCDSA, GUDSA, and GSDSA.

The dynamic storage areas are made from virtual storage pages taken from MVS storage subpools. In the dynamic storage areas, CICS arranges the storage in CICS subpools. The subpools are dynamically acquired as needed, a page at a time, from within the dynamic storage area. The storage that individual subpools use is shown in the domain subpool statistics in the CICS storage manager statistics.

CICS manages DSA storage in extents. An individual DSA consists of one or more extents.
- 24-bit storage extents are usually allocated in multiples of 256 KB. However, when transaction isolation is in operation, the UDSA is allocated in 1 MB extents.
- 31-bit storage extents are allocated in multiples of 1 MB.
- 64-bit storage extents are allocated in multiples of 1 GB.

Only the owning DSA can use an allocated extent, and a given extent cannot be shared between more than one DSA simultaneously.

The storage for the DSAs can be allocated from CICS-key storage, user-key storage, or read-only key-0 protected storage. The type of storage that is allocated for each DSA can depend on the settings for the **STGPROT** and **RENTPGM** system initialization parameters for the CICS region.

- The storage for the CDSA, ECDSA, ETDSA and GCDSA is always allocated from CICS-key storage.
- The **STGPROT** system initialization parameter specifies whether you want storage protection to operate in the CICS region.

  When you specify **STGPROT=YES**, or allow the system initialization parameter to default, the storage for the CICS dynamic storage areas for user applications is allocated from user-key storage. These DSAs are the UDSA, SDSA, EUDSA, ESDSA, GUDSA, and GSDSA. If you specify **STGPROT=NO**, the storage for these DSAs is allocated from CICS-key storage.
- The **RENTPGM** parameter specifies whether CICS allocates the read-only DSAs from read-only key-0 protected storage.

  When you specify **RENTPGM=PROTECT**, the read-only DSAs are allocated from read-only key-0 protected storage. These DSAs are the RDSA and the ERDSA. If you specify **RENTPGM=NOPROTECT**, the storage for these DSAs is allocated from CICS-key storage.

A dynamic storage area that is too small results in increased program compression or, more seriously, short-on-storage (SOS) conditions. You can examine the pressure on virtual storage by using the CICS storage manager statistics, which report the number of times that CICS went short on storage.

Related concepts:

"Short-on-storage conditions in dynamic storage areas" on page 105
If the limit for a dynamic storage area (DSA) is too small, the CICS region periodically enters a short-on-storage condition. Where possible, CICS curtails system activity until it can recover enough storage to resume normal operations. Use CICS messages and statistics to monitor when a short-on-storage (SOS) condition is entered, and when it is relieved.

Related information:

STGPROT system initialization parameter in Reference -> System definition

RENTPGM system initialization parameter in Reference -> System definition

DSAs in 24-bit storage: CDSA, UDSA, SDSA, and RDSA

The CICS dynamic storage areas (DSAs) below the line (below 16 MB) are in 24-bit storage. These storage areas do not have a collective name. The CICS system initialization parameter **DSALIM** specifies the limit on the total size of these dynamic storage areas.

The amount of storage specified by the **DSALIM** value is allocated as guaranteed storage at system initialization. Within this storage, CICS manages the following dynamic storage areas automatically, and you do not need to specify their individual sizes:

CDSA (CICS DSA)
: The storage area for all non-reentrant CICS-key RMODE(24) programs, all CICS-key task-lifetime storage in 24-bit storage, and for CICS control blocks that reside in 24-bit storage. The CDSA is always allocated from CICS-key storage.

UDSA (User DSA)
: The storage area for all user-key task-lifetime storage in 24-bit storage. If you specify the system initialization parameter **STGPROT=YES** for the CICS region, which is the default, the UDSA is allocated from user-key storage. If you specify **STGPROT=NO**, the UDSA is allocated from CICS-key storage.

SDSA (Shared DSA)

The storage area for any non-reentrant user-key RMODE(24) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for 24-bit storage with the SHARED option. If you specify the system initialization parameter **STGPROT=YES** for the CICS region, which is the default, the SDSA is allocated from user-key storage. If you specify **STGPROT=NO**, the SDSA is allocated from CICS-key storage.

RDSA (Read-only DSA)

The storage area for all reentrant programs and tables in 24-bit storage. If you specify the system initialization parameter **RENTPGM=PROTECT** for the CICS region, which is the default, the RDSA is allocated from read-only key-0 protected storage. If you specify **RENTPGM=NOPROTECT**, the RDSA is allocated from CICS-key storage.

DSAs in 31-bit storage: ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA

The group of CICS dynamic storage areas above the line (above 16 MB but below 2 GB) are collectively called the extended dynamic storage area (EDSA). This storage is 31-bit storage. The CICS system initialization parameter **EDSALIM** specifies the limit on the total size of these dynamic storage areas.

The amount of storage specified by the **EDSALIM** value is allocated as guaranteed storage at system initialization. Within this storage, CICS manages the following dynamic storage areas automatically, and you do not need to specify their individual sizes:

ECDSA (Extended CICS DSA)

The storage area for all non-reentrant CICS-key RMODE(ANY) programs, all CICS-key task-lifetime storage in 31-bit storage, and CICS control blocks that reside in 31-bit storage. The ECDSA is always allocated from CICS-key storage.

EUDSA (Extended user DSA)

The storage area for all user-key task-lifetime storage in 31-bit storage (above the line). If you specify the system initialization parameter **STGPROT=YES** for the CICS region, which is the default, the EUDSA is allocated from user-key storage. If you specify **STGPROT=NO**, the EUDSA is allocated from CICS-key storage.

ESDSA (Extended shared DSA)

The storage area for any non-reentrant user-key RMODE(ANY) programs, and also for any storage obtained by programs issuing CICS GETMAIN commands for 31-bit storage with the SHARED option. If you specify the system initialization parameter **STGPROT=YES** for the CICS region, which is the default, the ESDSA is allocated from user-key storage. If you specify **STGPROT=NO**, the ESDSA is allocated from CICS-key storage.

ERDSA (Extended read-only DSA)

The storage area for all reentrant programs and tables in 31-bit storage. If you specify the system initialization parameter **RENTPGM=PROTECT** for the CICS region, which is the default, the ERDSA is allocated from read-only key-0 protected storage. If you specify **RENTPGM=NOPROTECT**, the ERDSA is allocated from CICS-key storage.

ETDSA (Extended trusted DSA)

The storage area for any security-related CICS control blocks that reside in 31-bit storage. The ETDSA is always allocated from CICS-key storage.

DSAs in 64-bit storage: GCDSA, GUDSA, and GSDSA

CICS dynamic storage areas above the bar are collectively called the above-the-bar dynamic storage area (GDSA). This storage is 64-bit storage. The z/OS **MEMLIMIT** parameter limits the 64-bit storage in the CICS region, including the GDSA.

The **MEMLIMIT** value that the z/OS operating system assigns to the CICS address space controls the upper limit for 64-bit storage in the CICS region. This 64-bit storage includes both the GDSA, and MVS storage in the CICS region outside the GDSA.

In contrast, the 24-bit storage specified by the **DSALIM** value and the 31-bit storage specified by the **EDSALIM** value relate only to the CICS DSAs.

The GDSA does not preallocate an amount of guaranteed storage. The GDSA contains the following dynamic storage areas:

GCDSA (above-the-bar CICS DSA)
: The storage area for all CICS-key task-lifetime storage in 64-bit (above-the-bar) storage, and for CICS facilities that use 64-bit storage. See “CICS facilities that use 64-bit storage” on page 101. The GCDSA is always allocated from CICS-key storage.

GUDSA (above-the-bar user DSA)
: The storage area for all user-key task-lifetime storage in 64-bit (above-the-bar) storage. If you specify the system initialization parameter STGPROT=YES for the CICS region, which is the default, the GUDSA is allocated from user-key storage. If you specify STGPROT=NO, the GUDSA is allocated from CICS-key storage.

GSDSA (above-the-bar shared DSA)
: The storage area for any storage that programs obtain by issuing a CICS GETMAIN64 command to obtain 64-bit storage with the SHARED option. If you specify the system initialization parameter STGPROT=YES for the CICS region, which is the default, the GSDSA is allocated from user-key storage. If you specify STGPROT=NO, the GSDSA is allocated from CICS-key storage.

Storage protection

CICS uses the storage protection facilities that are available in the operating system to prevent CICS code and control blocks from being overwritten accidentally by your user application programs. To do this, separate dynamic storage areas (DSAs), with separate storage keys, are allocated for your user application programs, and for CICS code and control blocks. Access to a storage area is not permitted unless the access key matches the key for that storage area.

The storage allocated for most CICS code and control blocks is known as CICS-key storage, and the storage allocated for your user application programs is known as user-key storage.

In addition to CICS-key and user-key storage, CICS also uses key-0 storage for separate dynamic storage areas called the read-only DSAs (RDSA and ERDSA). The ERDSA is used for eligible re-entrant CICS and user application programs that are link-edited with the RENT and RMODE(ANY) attributes. The RDSA is used for eligible reentrant CICS and user application programs that are link-edited with the RENT and RMODE(24) attributes. The allocation of key-0 storage for the read-only DSAs is from the same storage limit as the other DSAs, as specified by the **DSALIM** and **EDSALIM** system initialization parameters.

Use of the storage protection facilities is optional. You can enable the facilities by using options on the system initialization parameters that are related to storage protection. Between them, you can use these parameters to define or control the following items:

- The storage key for the common work area (**CWAKEY**)
- The storage key for the terminal control table user areas (**TCTUAKEY**)
- A storage protection global option (**STGPROT**)
- A read-only program storage key option (**RENTPGM**)
- A transaction isolation option (**TRANISO**)

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; that is, a pure terminal-owning region (TOR) or a pure file-owning region (FOR) (where no distributed program link (DPL) requests are function-shipped).

The common work area (CWA):

The common work area (CWA) is an area of storage in your CICS region that any user application can access. You determine the size of this work area by using the **WRKAREA** system initialization parameter; you can specify sizes up to 3584 bytes.

If you omit the **WRKAREA** parameter, CICS allocates a 512-byte CWA by default. You specify the storage key for the CWA on the **CWAKEY** parameter.

This work area is available to all transactions in a CICS region, so you must ensure that the storage key is appropriate to the use of the CWA by all transactions. If any of the transactions run in user key and require write access, you must specify user-key storage for the CWA, otherwise such transactions fail with a storage protection exception (an ASRA abend). CICS obtains user-key storage for the CWA by default, and you must review the use of this storage by all programs before you decide to change it to CICS-key storage.

It is possible that you might want to protect the CWA from being overwritten by applications that should not have write access. In this case, provided all the applications that legitimately require write access to the CWA run in CICS key, you can specify CICS-key storage for the CWA.

Related information:

CWAKEY system initialization parameter in Reference -> System definition

WRKAREA system initialization parameter in Reference -> System definition

The terminal control table user areas:

A terminal control table user area (TCTUA) is an optional storage area associated with a terminal control table terminal entry (TCTTE). TCTUAs are available for application program use. You specify the storage key for TCTUAs globally for a CICS region by using the **TCTUAKEY** system initialization parameter.

By default, CICS obtains user-key storage for all TCTUAs. Review the use of TCTUAs in your CICS regions, and only specify CICS key for TCTUAs when you are sure that this is justified. If you specify CICS-key storage for TCTUAs, no user-key applications can write to any TCT user areas.

For SNA LUs, you specify that you want a TCTUA by means of the USERAREALEN attribute on the TYPETERM resource definition. The USERAREALEN attribute determines the TCTUA sizes for all terminals that reference the TYPETERM resource definition.

For sequential terminals, definitions are added to the terminal control table (TCT), and sizes are defined by means of the **TCTUAL** parameter on the DFHTCT TYPE=TERMINAL and TYPE=LINE entries. For information about the TCTUAL parameter, see Terminal control table (TCT) in Reference -> System definition.

Related information:

TCTUAKEY system initialization parameter in Reference -> System definition

The storage protection global option:

You can control whether your CICS region uses storage protection by specifying the **STGPROT** system initialization parameter. By default, CICS uses storage protection.

When you specify **STGPROT=YES**, or allow the system initialization parameter to default, the storage for the CICS dynamic storage areas for user applications is allocated from user-key storage. These DSAs are the UDSA, SDSA, EUDSA, ESDSA, GUDSA, and GSDSA. If you specify **STGPROT=NO**, the storage for these DSAs is allocated from CICS-key storage.

Running a CICS region without storage protection (**STGPROT=NO**) is suitable for pure terminal-owning regions (TORs) that do not execute user transactions.

Related information:

STGPROT system initialization parameter in Reference -> System definition

The transaction isolation global option:

CICS transaction isolation builds on CICS storage protection, enabling user transactions to be protected from one another. You can specify transaction isolation globally for a CICS region using the **TRANISO** system initialization parameter.

In addition to being able to specify the storage and execution key individually for each user transaction, you can specify that CICS is to isolate the user-key task-lifetime storage of a transaction to provide transaction-to-transaction protection. You do this by using the ISOLATE option of the TRANSACTION resource definition.

Transaction isolation does not apply to 64-bit storage.

Related information:

TRANISO system initialization parameter in Reference -> System definition

TRANSACTION attributes in Reference -> System definition

The read-only storage override option:

CICS obtains storage for the read-only DSAs (RDSA and ERDSA) from MVS read-only storage. You can override the selection of read-only storage for the RDSA and ERDSA by specifying NOPROTECT on the **RENTPGM** system initialization parameter.

The CICS loader automatically loads eligible modules into the RDSA and ERDSA; that is, if they are link-edited with the RENT attribute, and for the ERDSA with RMODE(ANY). You can specify **RENTPGM=NOPROTECT** if you do not want such modules to be loaded into read-only storage, perhaps because you are using a development aid package that sets break points in your application programs. When you specify **RENTPGM=NOPROTECT**, CICS still allocates separate read-only DSAs, but obtains CICS-key storage for the RDSA and ERDSA instead of read-only storage.

The **RENTPGM=NOPROTECT** override is only appropriate for development regions. In production CICS regions, **RENTPGM=PROTECT** provides the right level of protection for modules in the RDSA and ERDSA.

Related information:

RENTPGM system initialization parameter in Reference -> System definition

Setting the limits for CICS storage

Use the z/OS **REGION** and **MEMLIMIT** parameters to set limits for the storage for the CICS region. Use the CICS **DSALIM** and **EDSALIM** system initialization parameters to limit the total storage for the CICS dynamic storage areas (DSAs) in 24-bit and 31-bit storage.

About this task

The z/OS **REGION** and **MEMLIMIT** parameters specify the amount of virtual storage for the address space in which the CICS region runs.

- Use the z/OS **REGION** parameter to specify the amounts of 24-bit (below-the-line) and 31-bit (above-the-line) storage that are requested for the CICS region.
- Use the z/OS **MEMLIMIT** parameter to set the limit for the amount of 64-bit (above-the-bar) storage for the CICS region.

  The 64-bit storage specified by the **MEMLIMIT** value includes the CICS dynamic storage areas above the bar (the GDSA) and MVS storage in the CICS region outside the GDSA.

For more information about these parameters, see "CICS region size" on page 87.

The storage specified by the **REGION** value includes the storage specified by the CICS system initialization parameters **DSALIM** and **EDSALIM**. These parameters determine the overall limits within which CICS can allocate storage for the CICS DSAs.

- Use the **DSALIM** parameter to set overall limits for the CICS DSAs in 24-bit (below-the-line) storage.
- Use the **EDSALIM** parameter to set overall limits for the extended dynamic storage area (EDSA), that is, the CICS DSAs in 31-bit (above-the-line) storage.

CICS allocates individual dynamic storage areas automatically, and you do not need to specify their sizes. CICS varies the size of the individual dynamic storage areas as the need arises.

- CICS allocates DSAs in 24-bit storage within the limits set by **DSALIM**.
- CICS allocates DSAs in 31-bit storage within the limits set by **EDSALIM**.
- CICS allocates DSAs in 64-bit storage within the limits set by **MEMLIMIT**.

Procedure

- To estimate and change the setting of the z/OS parameter **REGION**, see "Estimating and setting REGION."
- To estimate and change the setting of the z/OS parameter **MEMLIMIT**, and to check its value in a running CICS system, see "Estimating, checking, and setting MEMLIMIT" on page 100.
- To estimate and change the setting of the CICS system initialization parameter **DSALIM**, see "Estimating, checking, and setting DSALIM" on page 96.
- To estimate and change the setting of the CICS system initialization parameter **EDSALIM**, see "Estimating, checking, and setting EDSALIM" on page 98.

Estimating and setting REGION

The z/OS **REGION** parameter limits the amount of 24-bit and 31-bit storage (storage below the bar) that the CICS address space can use. This value includes all the storage below the bar in the private area, except for a 16 KB system region in 24-bit storage, and the items in the high private area such as the LSQA.

About this task

For an explanation of the storage areas in the z/OS address space below 2 GB, see The Virtual Storage Address Space in the z/OS MVS Initialization and Tuning Guide.

You can request up to 2047 MB of storage below the bar for the CICS region, but you must ensure that you leave enough storage below the bar for MVS to use in the high private area. The items in the high private area are the local shared queue area (LSQA), scheduler work area (SWA), and subpools 229 and 230. These items exist in both 24-bit (below-the-line) storage and 31-bit (above-the-line) storage. The LSQA in 31-bit storage is called the extended LSQA. Some of this storage is used for control blocks, and some is used by z/OS Communications Server and other programs.

Within the value that you specify for **REGION**, the following types of storage are included:

- The CICS DSAs in 24-bit storage. The storage for these DSAs is limited by the **DSALIM** system initialization parameter.
- The CICS DSAs in 31-bit storage. The storage for these DSAs is limited by the **EDSALIM** system initialization parameter.
- Storage used by the CICS kernel.
- MVS storage obtained by MVS GETMAIN requests.
- CICS dispatcher
- CICS storage manager
- CICS lock manager

You cannot alter the **REGION** value for the CICS region while CICS is running. You can specify a new value on the next start of the CICS region.

Procedure

1. To determine the maximum value for the **REGION** parameter:
   a. Use RMF or another storage monitor to determine the size of your private area.
   b. Apply the following formula:

```
Maximum possible REGION =
   Size of private area
   - Size of system region (16K)
   - (LSQA + SWA + subpools 229 and 230)
```

For more information about the high private area and LSQAs, and estimates for the sizes of the items in the high private area, see "High private area" on page 134.

c. For safety, do not use more than 80% or 90% of this maximum value for the **REGION** parameter. It is useful to maintain some free storage between the top of the CICS region and the bottom of the high private area. If the system is static or does not change much, use up to 90% of this number. If the system is dynamic, or changes frequently, 80% would be better.

2. To estimate the value that you require for the **REGION** parameter to meet your storage needs, add up your estimates for the following areas of storage:
   - The area of 24-bit storage for CICS DSAs below the line, specified by the **DSALIM** system initialization parameter. See "Estimating, checking, and setting DSALIM."
   - The area of 31-bit storage for CICS DSAs above the line (the EDSA), specified by the **EDSALIM** system initialization parameter. See "Estimating, checking, and setting EDSALIM" on page 98.
   - The small amount of storage used by the CICS kernel outside the CICS DSAs. See "CICS kernel storage" on page 128.
   - MVS storage obtained by MVS GETMAIN requests outside the CICS DSAs. See "MVS storage below 2 GB" on page 130.
3. For instructions to specify the **REGION** parameter, and information about the amount of storage that z/OS allocates in response to your request, see REGION Parameter in the z/OS MVS JCL Reference. You cannot alter the **REGION** value for a running CICS region. You can set **REGION** in the following ways:
   - You can specify the **REGION** parameter in the JOB statement in the CICS JCL. In this situation, each step of the job runs in the requested amount of space.
   - You can specify the **REGION** parameter in the EXEC statement (program execution line) for CICS. In this situation, each step runs in its own amount of space. Use the EXEC statement instead of the JOB statement if different steps need greatly different amounts of space. For example, you could use the EXEC statement if you are using extra job steps to print auxiliary trace data sets after CICS has shut down (as in the DFHIVPOL installation verification procedure).
   - The z/OS installation exit IEFUSI can limit the **REGION** value that you specify. For information about IEFUSI, see IEFUSI - Step Initiation Exit in z/OS MVS Installation Exits.

Estimating, checking, and setting DSALIM

The **DSALIM** system initialization parameter specifies the upper limit of the total amount of storage within which CICS can allocate the individual dynamic storage areas (DSAs) that reside in 24-bit storage (below 16 MB, also known as below the line). If your installation is constrained for 24-bit storage, set a value for the **DSALIM** parameter equivalent to the sum of the CDSA and UDSA. If you have sufficient virtual storage to allow a greater value for **DSALIM**, you can use the formulas given here.

About this task

Accurate sizing of the **DSALIM** value is not critical. It is better to specify a **DSALIM** value that is slightly greater than your expected requirements rather than slightly smaller. You can tune the **DSALIM** parameter to a smaller value after you obtain data from your running system.

Make sure that you understand the requirements for MVS storage in 24-bit storage outside the CICS DSAs, to avoid other subsystem problems. For more information about the operating system components that use MVS storage, see "MVS storage below 2 GB" on page 130.

The minimum **DSALIM** value is 2 MB and the default value is 5 MB. The maximum **DSALIM** value is 16 MB. The extent size for the CDSA, RDSA, and SDSA is in 256 KB increments. If transaction isolation is active, the extent size for the UDSA is 1 MB and each UDSA extent must be aligned on a megabyte boundary. If transaction isolation is not active, the allocation is in 256 KB extents.

CICS allocates 24-bit kernel stack storage when it is required. Tasks obtain 4 KB extension stack segments whenever they require 24-bit stack storage. CICS preallocates a reserve pool of 24-bit extension stack segments that tasks can use if no other 24-bit stack storage is available. For more information about kernel storage, see "CICS kernel storage" on page 128.

Procedure

- To check the **DSALIM** value that currently applies to a running CICS region, use one of the following methods:
  - CICS Explorer: **Global Dynamic Storage Areas** view
  - CICSPlex SM: **Dynamic storage areas - CICSDSA** view
  - CEMT: **CEMT INQUIRE DSAS** or **CEMT INQUIRE SYSTEM**
  - CICS SPI: **INQUIRE SYSTEM**
- To estimate a suitable **DSALIM** value, use the following steps:
  1. If you have sufficient virtual storage to specify a generous **DSALIM** value, use the following formula to estimate a value:

     CDSA + UDSA + SDSA + RDSA

     Round up the value of each component in your calculation to a 256 KB boundary.
  2. If your current installation **DSALIM** value is larger than necessary, use the following formula to estimate a **DSALIM** value:

     Peak CDSA Used + Peak UDSA Used + Peak SDSA Used + Peak RDSA Used

     Round up the value of each component in your calculation to a 256 KB boundary.
- To change the **DSALIM** value for the CICS region, use one of the following methods while CICS is running:
  - CICS Explorer: **Global Dynamic Storage Areas** view
  - CICSPlex SM: **Dynamic storage areas - CICSDSA** view
  - CEMT: **CEMT SET DSAS** or **CEMT SET SYSTEM**
  - CICS SPI: **SET SYSTEM**

Results

If there are no extents free in the CICS DSAs in 24-bit storage, CICS cannot implement a reduction of **DSALIM**. The storage manager applies MVS FREEMAIN requests to extents as they become available until the new **DSALIM** value is reached.

A short-on-storage condition can occur when you reduce **DSALIM**.

Estimating, checking, and setting EDSALIM

The **EDSALIM** system initialization parameter specifies the upper limit of the total amount of storage within which CICS can allocate the individual extended dynamic storage areas (EDSAs) that reside in 31-bit (above-the-line) storage. Set the value for the **EDSALIM** parameter as large as you can after consideration of other areas, especially MVS storage.

About this task

The maximum value that you can specify for **EDSALIM** is limited by the following factors:

- The size that you specified for the CICS region on the z/OS **REGION** parameter in the CICS job or procedure. The value of **EDSALIM** must be less that the value of **REGION**.
- The amount of MVS storage, outside the CICS DSAs, that you require to satisfy MVS GETMAIN requests for 31-bit storage. For more information about the operating system components that use MVS storage, see "MVS storage below 2 GB" on page 130.

Accurate sizing of the **EDSALIM** value is not critical. A good approach is as follows:

- Initially specify an **EDSALIM** value that is slightly greater than your expected requirements. The default setting, 800 MB, enables the CICS region to run a reasonable workload.
- Monitor the use of each CICS DSA in the EDSA while your system is running near peak loads.
- Tune your **EDSALIM** value in the running CICS system.

Try not to specify the largest possible **EDSALIM** value (for example, the maximum allowable region size). If you use the maximum possible limit, you might not receive any warnings about a shortage of virtual storage until the problem becomes difficult to resolve.

You can obtain information about your current EDSA use by looking at the CICS storage manager statistics. See the information about DSA sizes in the storage manager statistics, dynamic storage areas, and task subpools. Automatic DSA sizing removes the need for accurate storage estimates for individual DSAs, with CICS dynamically changing the size of DSAs as demand requires.

The minimum **EDSALIM** value is 48 MB, which is the minimum required to start a CICS region. The default **EDSALIM** value is 800 MB. The maximum **EDSALIM** size is 2047 MB, which is 2 GB minus 1 MB. The extent size for the ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA is 1 MB.

Kernel stack storage is also allocated from the EDSA. For more information about kernel storage see "CICS kernel storage" on page 128.

Remember that for CICS regions that run with transaction isolation active (set by using the **TRANISO** system initialization parameter), the transaction isolation facility increases the allocation of some virtual storage above 16 MB.

If transaction isolation is active, CICS allocates user-key task-lifetime storage above 16 MB in multiples of 1 MB (the minimum unit of storage allocation for the EUDSA when transaction isolation is active is 1 MB). However, MVS paging activity affects only the storage that is used (referenced), and unused parts of the 1 MB allocation are not paged.

For a CICS region that runs without transaction isolation, CICS allocates user-key task-lifetime storage above 16 MB in multiples of 64 KB.

The subspace group facility uses more real storage, because MVS creates a page and segment table from real storage for each subspace. The CICS requirement for real storage varies according to the transaction load at any one time. As a guideline, each task in the system requires 9 KB of real storage, and this should be multiplied by the number of concurrent tasks that can be in the system at any one time (governed by the **MXT** system initialization parameter).

Procedure

- To check the **EDSALIM** value that currently applies to a running CICS region, use one of the following methods:
  - CICS Explorer: **Global Dynamic Storage Areas** view
  - CICSPlex SM: **Dynamic storage areas - CICSDSA** view
  - CEMT: **CEMT INQUIRE DSAS** or **CEMT INQUIRE SYSTEM**
  - CICS SPI: **INQUIRE SYSTEM**
- To estimate a suitable **EDSALIM** value, use the following steps:
  1. Check the **MXT** value for your CICS region. The **MXT** system initialization parameter does not include CICS system tasks, and it might also be set to a value larger than necessary. The safest estimate for calculating an **EDSALIM** value is to assume **MXT** as the number of concurrent active tasks.
  2. Check the setting of the **TRANISO** system initialization parameter for your CICS region. For the EUDSA, if the **TRANISO** parameter is set to NO for the CICS region, allow 64 KB per concurrent active task. If the **TRANISO** parameter is set to YES, allow 1 MB per concurrent active task. If your applications use more than 64 KB per task with the **TRANISO** parameter set to NO, or more than 1 MB per task with the **TRANISO** parameter set to YES, adjust the formulas accordingly. If you adjust the formulas, use multiples of 64 KB or 1 MB.
  3. If you have sufficient virtual storage to specify a generous **EDSALIM** value, use one of the following formulas to estimate a value. Round up the value of each component in your calculation to a 1 MB boundary, the size of an extent, to allow for fragmentation and partially used extents.

     For TRANISO=NO:
     ECDSA + EUDSA + ESDSA + ERDSA + ETDSA + (64 K x MXT)

     For TRANISO=YES:
     ECDSA + EUDSA + ESDSA + ERDSA + ETDSA + (1 MB x MXT)
  4. If your current installation **EDSALIM** and **MXT** values are larger than necessary, use one of the following formulas to estimate an **EDSALIM** value. Round up the value of each component in your calculation to a 1 MB boundary, the size of an extent, to allow for fragmentation and partially used extents.

For TRANISO=NO:
Peak ECDSA Used + Peak EUDSA Used + Peak ESDSA Used + Peak ERDSA Used + Peak ETDSA Used - EUDSA Peak Page Storage in Task Subpools + (64 K x Peak number of tasks)

For TRANISO=YES:
Peak ECDSA Used + Peak EUDSA Used + Peak ESDSA Used + Peak ERDSA Used + Peak ETDSA Used - EUDSA Peak Page Storage in Task Subpools + (1 MB x Peak number of tasks)

- To change the **EDSALIM** value for the CICS region, use one of the following methods while CICS is running:
  - CICS Explorer: **Global Dynamic Storage Areas** view
  - CICSPlex SM: **Dynamic storage areas - CICSDSA** view
  - CEMT: **CEMT SET DSAS** or **CEMT SET SYSTEM**
  - CICS SPI: **SET SYSTEM**

Results

If you under-specify **EDSALIM**, your system can go short on storage and you might not be able to issue CICS commands to increase the limit. In this situation, use the CICS Explorer or CICSPlex SM to increase the **EDSALIM** value.

If there are no extents free in the CICS DSAs, CICS cannot implement a reduction of **EDSALIM**. The storage manager applies MVS FREEMAIN requests to extents as they become available until the new **EDSALIM** value is reached.

Estimating, checking, and setting MEMLIMIT

The z/OS **MEMLIMIT** parameter limits the amount of 64-bit (above-the-bar) storage that the CICS address space can use. This storage includes the CICS dynamic storage areas above the bar (collectively called the GDSA) and MVS storage in the CICS region outside the GDSA.

About this task

CICS requires a **MEMLIMIT** value of 6 GB; any additional use by applications or JVMs should be allowed for with a larger value of **MEMLIMIT**. If you attempt to start a CICS region with a **MEMLIMIT** value that is less than 6 GB, message DFHSM0602 is issued, a system dump with the dump code KERNDUMP is produced, and CICS terminates.

You cannot alter the **MEMLIMIT** value for the CICS region while CICS is running. You can specify a new **MEMLIMIT** value on the next start of the CICS region.

Procedure

- To check the **MEMLIMIT** value that currently applies to a running CICS region, use one of the following methods:
  - CICS Explorer: **Regions** view or **Global Dynamic Storage Areas** view
  - CICSPlex SM: **Dynamic storage areas - CICSDSA** view or **CICS regions - CICSRGN** view
  - CEMT: **CEMT INQUIRE DSAS** or **CEMT INQUIRE SYSTEM**
  - CICS SPI: **INQUIRE SYSTEM**
- To estimate a suitable **MEMLIMIT** value for a CICS region, add up the storage requirements for those facilities that use 64-bit storage that you use in your CICS region.

For a list of CICS facilities that use 64-bit storage in the CICS region and the relevant storage subpools, see CICS facilities that can use 64-bit storage in Improving performance.

The CICS storage manager statistics show the storage used in each CICS subpool in the GDSA in a running CICS region. For information about the subpools in the GDSA, see CICS subpools in the GCDSA in Improving performance and CICS subpools in the GSDSA in Improving performance.

- To change a **MEMLIMIT** value in z/OS, or determine the value that applies to a CICS region that you are setting up, see Limiting the use of memory objects in the z/OS MVS Programming: Extended Addressability Guide. **MEMLIMIT** can be set in one of the following ways:
  - A **MEMLIMIT** value can be specified in the JOB statement in the CICS JCL, or in the EXEC statement (program execution line) for CICS.
  - If there is no **MEMLIMIT** value specific to the CICS region, the **MEMLIMIT** value that is set in the z/OS SMFPRM*xx* PARMLIB member, or the system default, applies.
  - The z/OS installation exit IEFUSI can override any other **MEMLIMIT** values.

  The following example shows a **MEMLIMIT** value set in the program execution line:

```
//CICS EXEC PGM=DFHSIP,PARM='SI',REGION=0M,MEMLIMIT=6G
```

CICS facilities that use 64-bit storage:

The CICS facilities that use 64-bit storage and the amount of storage that each facility requires are listed. You can use this information when you estimate a suitable **MEMLIMIT** value for your CICS region.

The following table lists CICS facilities that use 64-bit storage and the relevant CICS storage subpools, and indicates the amount of storage that each facility requires. You can identify the facilities that you use in your CICS region that require 64-bit storage. You can then examine the storage requirement for those facilities and use this information to estimate a suitable value for the z/OS **MEMLIMIT** parameter.

Table 4. CICS facilities that use 64-bit storage

CICS facility	Storage use	CICS subpool for storage	Notes®
Application context data control blocks	220 bytes for each program that is defined as an application entry point.	PGPPTE64	
Association data control blocks	Approximately 1 KB for each active task.	MN_ADCS	
CICS management client interface (CMCI)	For the details to estimate storage requirements, see Estimating storage requirements for CMCI.	WU_64	The storage is used for retained results and metadata.
CICSPlex SM API result sets	The size of the result set depends on the application.	CPSM_64	For information about result sets, see Working with result sets in Developing system programs.
Console queue processing	1 MB per subpool	CQCQ_TR CQCQ_XT	

Table 4. CICS facilities that use 64-bit storage (continued)

CICS facility	Storage use	CICS subpool for storage	Notes®
Containers and channels	Limited to 5% of the **MEMLIMIT** value per transaction.	PGCSDB	The storage remains in use until the channel goes out of scope, or an application deletes the container or the channel.
Event processing		EP_64	The storage is used for control blocks for items in event capture queues.
Internal trace table	Minimum 16 KB Maximum 1 GB Default 12288 KB (12 MB) Controlled by the **TRTABSZ** system initialization parameter.	MVS storage outside the CICS DSAs	
JVM servers	Add up the following values for each JVM server in the CICS region: • **-Xmx** value in the JVM profile • HEAP64 value in DFHAXRO • LIBHEAP64 value in DFHAXRO • STACK64 value in DFHAXRO multiplied by number of allowed threads	MVS storage outside the CICS DSAs	The STACK64 value is multiplied by the number of threads that are allowed in the JVM server. To calculate the number of allowed threads, add the THREADLIMIT attribute value on the JVMSERVER resource to the value of the **-Xgcthreads** parameter. This Java parameter controls the number of garbage collection helper threads in the JVM. The minimum value of the **-Xgcthreads** parameter is 1, and the default is the number of physical CPUs present minus 1.
Loader control blocks	Variable	LD_APES LD_CPES LD_CSECT	The storage used depends on the number of program load operations in the CICS region.
Loader control blocks for installed application elements (IAE)	368 bytes for each IAE	LD_IAES	Each installed application element represents a particular version of an application that is deployed on a platform.

Table 4. CICS facilities that use 64-bit storage (continued)

CICS facility	Storage use	CICS subpool for storage	Notes®
Main temporary storage	Minimum 1 MB Maximum 32 GB Default 64 MB Controlled by the **TSMAINLIMIT** system initialization parameter.	TSDTN TSMAIN TSMN0064 TSMN0128 TSMN0192 TSMN0256 TSMN0320 TSMN0384 TSMN0448 TSMN0512 TSMN0576 TSMN0640 TSMN0704 TSMN0768 TSMN0832 TSMN0896 TSMN0960 TSMN1024 TSQUEUE TSTSI	**TSMAINLIMIT** specifies the maximum storage that can be used, and is limited to 25% of the **MEMLIMIT** value. The CICS statistics show actual use.
Managed Platform	640 bytes for each user task	DFHMPMDR DFHMPPMB DFHMPTAS	Storage in the DFHMPMDR and DFHMPPMB subpools is used for control blocks for installed policies. It remains in use until the bundle that defines the policy is disabled. Storage in the DFHMPTAS subpool contains the policy counters for a task. It is allocated for every user task if **any** polices are installed in the CICS region. The storage is freed at task end.
Message tables	Minimum 3 MB for the message modules in English. Add 1 MB if the user message table is loaded. Add 3 MB for each additional language that is loaded.	MVS storage outside the CICS DSAs	Message modules in English are always loaded.
Storage allocation control blocks	Variable	MVS storage outside the CICS DSAs	Storage used depends on the amount of storage allocation activity in your system; for example, more storage is required for subpools that keep an element chain and that have many small records.
Transaction dump trace table	Minimum 16 KB Maximum 1 GB Default 1024 KB Controlled by the **TRTRANSZ** system initialization parameter.	MVS storage outside the CICS DSAs	CICS obtains this storage only when a transaction dump is produced.

Table 4. CICS facilities that use 64-bit storage (continued)

CICS facility	Storage use	CICS subpool for storage	Notes®
z/OS XML System Services (XMLSS) parser I/O buffers		ML64GNRL	A number of facilities, including CICS web services, use the parser for XML parsing.

Related information:

Calculating storage requirements for JVM servers in Improving performance

TRANISO system initialization parameter in Reference -> System definition

TRTRANSZ system initialization parameter in Reference -> System definition

TSMAINLIMIT system initialization parameter in Reference -> System definition

DSA size limits

It is not advisable to set the size of individual dynamic storage areas (DSAs), and usually it is not necessary. However, it is possible to set the size of some DSAs by using the **CDSASZE**, **UDSASZE**, **RDSASZE**, **ECDSASZE**, **EUDSASZE**, **ESDSASZE**, and **ERDSASZE** system initialization parameters.

For example, **CDSASZE** sets the size of the CICS dynamic storage area (CDSA), and **ECDSASZE** specifies the size of the extended CICS dynamic storage area (ECDSA). The default value for these parameters is 0, indicating that the size of the DSA can change dynamically. If you specify a nonzero value, the DSA size is fixed.

If you specify DSA size values that in combination do not allow sufficient space for the remaining DSAs, CICS fails to initialize.

- The limit on the storage available for the DSAs in 24-bit storage (below 16 MB) is specified by the **DSALIM** system initialization parameter. You must allow at least 256K for each DSA in 24-bit storage for which you have not set a size.
- The limit on the storage available for the DSAs in 31-bit storage (above 16 MB but below 2 GB) is specified by the **EDSALIM** system initialization parameter. You must allow at least 1 MB for each DSA in 31-bit storage for which you have not set a size.

You cannot set the size of individual DSAs in 64-bit storage; that is, in the above-the-bar DSA (GDSA).

Related information:

CDSASZE system initialization parameter in Reference -> System definition

UDSASZE system initialization parameter in Reference -> System definition

RDSASZE system initialization parameter in Reference -> System definition

ECDSASZE system initialization parameter in Reference -> System definition

EUDSASZE system initialization parameter in Reference -> System definition

ESDSASZE system initialization parameter in Reference -> System definition

ERDSASZE system initialization parameter in Reference -> System definition

Coding conventions for DSA limits

You can specify the size of the DSA limits as a number of bytes, a number of kilobytes, or a number of megabytes.

Use the letter K as a suffix to indicate that the value represents a whole number of kilobytes. Use the letter M as a suffix to indicate that the value represents a whole number of megabytes. For example, 2 MB can be coded as either `2048K` or `2M`. (1 KB = 1024 bytes; 1 MB = 1024 KB = 1048576 bytes.)

If the value you specify is not a multiple of 256 KB for **DSALIM**, or 1 MB for **EDSALIM**, CICS rounds up the value to the next multiple.

You cannot specify fractions of megabytes; you must code sizes in bytes or kilobytes. Some examples are shown in Table 5.

Table 5. Examples of DSA limit values in bytes, kilobytes, and megabytes

Coded as:					
bytes	2097152	3145788	3670016	4194304	4718592
kilobytes	2048K	3072K	3584K	4096K	4608K
megabytes	2M	3M	-	4M	-

Short-on-storage conditions in dynamic storage areas

If the limit for a dynamic storage area (DSA) is too small, the CICS region periodically enters a short-on-storage condition. Where possible, CICS curtails system activity until it can recover enough storage to resume normal operations. Use CICS messages and statistics to monitor when a short-on-storage (SOS) condition is entered, and when it is relieved.

CICS attempts to resolve pressures on storage before entering a short-on-storage condition. When CICS starts to become short on space in a DSA, the situation is known as a storage stress condition. Where possible, CICS takes actions such as deleting programs that are not being used (program compression), deleting cached copies of any document templates, and searching for free extents in other DSAs. If these actions fail to resolve the storage stress condition, CICS declares an SOS condition for the DSA.

During an SOS condition, CICS takes steps to limit work, so that there is enough storage to process work that is already in progress. CICS prevents acquisition of new input message areas, and defers all ATTACH requests from CICS system modules. Limiting work degrades the performance of the CICS region. In extreme circumstances, an SOS condition might also lead to storage deadlock abends.

When an SOS condition is entered, one of the following messages is issued:
- DFHSM0131 for 24-bit storage
- DFHSM0133 for 31-bit storage
- DFHSM0606 for 64-bit storage

SOS conditions are also recorded in the CICS statistics for the dynamic storage area ("Times went short on storage"). You can use the CICS commands **CEMT INQUIRE SYSTEM**, **EXEC CICS INQUIRE SYSTEM**, and **CEMT INQUIRE DSAS** to inquire about SOS conditions.

When you observe an SOS condition, first determine whether the affected storage is 24-bit, 31-bit, or 64-bit.
- For 24-bit storage, check whether the limit for the DSAs in 24-bit storage is as high as possible. If required, you can change the **DSALIM** parameter while CICS is running.

- For 31-bit storage, check whether the limit for the extended dynamic storage area (EDSA) is as high as possible. If required, you can change the **EDSALIM** parameter while CICS is running.
- For 64-bit storage, check whether there is sufficient 64-bit storage for the CICS region. If required, you can change the z/OS **MEMLIMIT** value, but only on the next start of the CICS region.

For instructions to change these limits, see "Setting the limits for CICS storage" on page 94.

CICS reserves areas of contiguous virtual storage, called storage cushions, in each DSA. A storage cushion is used only when there is not enough free storage in the DSA to satisfy an unconditional GETMAIN request. In a storage stress condition, the storage cushion might avert a storage deadlock. The CICS storage manager statistics for the dynamic storage areas show the number of times that CICS needed to use storage from the cushion. A request might be larger than all the remaining storage in the DSA, so that even the storage in the cushion is insufficient. When a request is suspended for this reason, the suspension is also shown in the CICS storage manager statistics for the dynamic storage areas.

Short-on-storage conditions for 24-bit and 31-bit storage

When an individual DSA in 24-bit or 31-bit storage, for example the CDSA, requires additional storage, the CICS storage manager allocates another extent to that DSA. Additional extents can be acquired as necessary until the **DSALIM** or **EDSALIM** limit is reached, as appropriate. When all the possible extents are allocated, CICS searches for a free extent in another DSA, to relocate it to the DSA in need. For CICS to remove an extent from one DSA so that it can be allocated to another, all pages in the extent must be free. That is, no pages must be allocated to any subpool.

Program compression might be triggered when the **DSALIM** or **EDSALIM** limit is approached and there are few free or empty extents available. The DSAs that contain programs are evaluated individually to determine whether program compression is required. In systems with a moderate proportion of loadable programs, program compression is an indicator of pressure on virtual storage.

CICS considers a short-on-storage condition for a DSA in 24-bit or 31-bit storage if all the following circumstances apply:

- No further extents can be allocated or relocated from other DSAs.
- Program compression has been attempted.
- All nonresident programs that are suitable for deletion, and that are not in use, have been deleted.
- All cached copies of document templates that are suitable for deletion. See Caching and refreshing of document templates in Developing applications.
- Storage from the storage cushion is in use (that is, the number of free pages is less than the number of pages in the cushion), or at least one request is suspended because there is no contiguous area of storage large enough for it, or both of these conditions apply.

Short-on-storage conditions for 64-bit storage

For 64-bit storage, CICS tracks the total amount of 64-bit storage in use for the CICS address space. This storage includes both the above-the-bar dynamic storage area (GDSA), and MVS storage in the CICS region outside the GDSA.

CICS considers an SOS condition when storage from the storage cushion is in use, or at least one request is suspended because there is no contiguous area of storage large enough for it, or both of these conditions apply. No further extents can be allocated for a DSA in 64-bit storage if the sum of all allocated above-the-bar storage and the size of a new extent would exceed the **MEMLIMIT** value.

The CICS storage manager statistics show 64-bit storage usage. The CICS storage manager dynamic storage areas statistics show storage usage for the DSAs in the GDSA. Statistics of interest include the following:
- Current GDSA active
- Peak GDSA active
- Number of IARV64 CONVERT(FROMGUARD) failures
- Current GDSA allocated
- Peak GDSA allocated
- Times cushion released
- Times went short on storage

An IARV64 CONVERT(FROMGUARD) failure indicates that a request for 64-bit storage has failed. A request might fail because there is not enough auxiliary storage in the system to back the request. Also, a request might fail because a component that the CICS storage manager does not control, for example, a JVM server, has allocated so much storage that the storage manager is affected. CICS cannot resolve pressures on storage caused by components outside the GDSA allocating storage, so you must use the CICS statistics to identify such problems.

Avoiding short-on-storage conditions

To optimize your use of the CICS dynamic storage areas and their storage cushions, and help to avoid short-on-storage conditions, follow these principles.

Procedure

- The lower the number of concurrent transactions in the system, the lower the usage of virtual storage. If you can improve the internal response time for transactions, for example by minimizing physical I/O, you can decrease the usage of virtual storage.
- Avoid making large GETMAIN requests in your application programs. The storage cushion might not be large enough to satisfy a request for a large contiguous block of storage.
- Define programs as resident only where necessary. CICS cannot delete resident programs to reclaim space in a DSA, even if the programs are not in use.
- Use the CICS storage manager statistics to monitor storage cushion releases and storage request suspensions. If these incidents occur frequently, investigate the cause. If necessary, reduce the maximum number of user tasks (using the **MXT** system initialization parameter) to reduce the number of tasks that use main storage.
- Try to define a reasonable number of transactions as SPURGE(YES) and with a DTIMOUT value. Only transactions defined in this way can be purged during an SOS condition, if they have been waiting for storage for longer than the DTIMOUT value. If there are too few purgeable transactions, storage might become deadlocked in the CICS system.

Analyzing short-on-storage conditions

Analysis of short-on-storage (SOS) problems begins by obtaining a dump when the system is in an SOS condition.

Procedure

1. Set an entry in the dump table to produce a dump when message DFHSM0131, DFHSM0133, or DFHSM0606 is issued. For example, to produce a dump the first time message DFHSM0131 is issued, use the following command:
   ```
   CEMT SET SYDUMPCODE(SM0131) SYSDUMP MAXIMUM(1) ADD
   ```
2. When you obtain the dump, enter the following IPCS commands:
   a. Use the IPCS command `VERBX CICS700 'SM=3'` to format the SM control blocks.
   b. Use the IPCS command `VERBX CICS700 'LD=3'` to format the LD control blocks.
3. Run DFH0STAT just before the statistics interval completes. For example, if the statistics interval is 1 hour, run DFH0STAT at 59 minutes. DFH0STAT provides useful information in the storage summary without a breakdown by subpool. See "The sample statistics program, DFH0STAT" on page 423 for more information.
4. In the information that you have collected, examine the DSA summaries, noting which DSAs are short on storage and the amount of free space in the other DSAs. The amount of free space 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. Also, look for evidence of large amounts of redundant program storage (RPS), which can cause a short-on-storage condition. Redundant program storage can be identified in the domain subpool summary and the loader domain summary.

Example

The dump extracts in this example are from a situation where the UDSA is short on storage.

Storage extents in 24-bit storage (below the line) are always allocated in multiples of 256 KB, except for the UDSA. If transaction isolation is active, the extent size for the UDSA is 1 MB, and each UDSA extent must be aligned on a megabyte boundary. If translation isolation is not active, the allocation is in 256 KB extents. You must allow for some fragmentation between the 256 KB extents of the CDSA, RDSA, and SDSA, compared with the 1 MB extents of the UDSA.

Storage extents in 31-bit storage (above the line) are allocated in multiples of 1 MB.

Storage extents in 64-bit storage (above the bar) are allocated in multiples of 2 GB.

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 free space. For example, the extent beginning at 00700000 running through 0073FFFF has only 4 KB allocated and 252 KB 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 that 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 PREVIOUS
   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 in which 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
```

What to do next

1. Review the storage limits for your CICS system. See “Setting the limits for CICS storage” on page 94.
2. For an SOS condition in 24-bit storage, determine whether the **DSALIM** parameter is set as large as possible. See “Estimating, checking, and setting DSALIM” on page 96.
3. For an SOS condition in 31-bit storage, determine whether the **EDSALIM** parameter is set as large as possible. See “Estimating, checking, and setting EDSALIM” on page 98.

4. For an SOS condition in 64-bit storage, determine whether the z/OS **MEMLIMIT** parameter is set to an appropriate value. See "Estimating, checking, and setting MEMLIMIT" on page 100.
5. Review the use of options such as the maximum task specification (**MXT** parameter) and defining programs as resident, to keep down the overall storage requirement. Changing these settings might limit task throughput. You can also reduce a storage constraint below 16 MB by using programs that run above 16 MB. In addition, using the LPA reduces the amount of storage used in LDNUCRO by approximately 100 KB.
6. Consider the tuning possibilities of z/OS and other tuning possibilities outside CICS. Also consider ways of dividing your CICS region.
7. Consider enabling the CICS self-tuning mechanism, or fixing the size of one or more individual DSAs by using the appropriate SIT overrides. For instructions, see "Fixing short-on-storage conditions caused by subpool storage fragmentation."

Fixing short-on-storage conditions caused by subpool storage fragmentation

You might experience short-on-storage conditions in 24-bit storage or 31-bit storage despite increasing the **DSALIM** or **EDSALIM** limits, respectively. In this situation, you might need to enable the CICS self-tuning mechanism. It is also possible to fix the size of each individual DSA by using the corresponding SIT override.

About this task

Use the self-tuning mechanism and the SIT overrides only if increasing the **DSALIM** or **EDSALIM** limit does not completely resolve the short-on-storage problems.

Allocating into managed extents can result in a block of storage in an extent that is insufficient to satisfy a GETMAIN request. With the dynamic nature of the subpools and DSAs, this situation will probably resolve as the extent storage is reused. If you specify the initial DSA size using the SIT override for the affected DSA, CICS reserves contiguous extents up to the amount specified, and eliminates the blocks of storage.

Tip: Define MAPS as MAPS. If you defining MAPS as programs, they are loaded into LDRES rather than into LDNUC. LDRES is part of the SDSA and is more sensitive to fragmentation.

Procedure

1. 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 manipulate subpool records using the CICS-supplied utility program, DFHSMUTL, see Local catalog storage program (DFHSMUTL) in the *CICS Operations and Utilities Guide*.
2. You can fix the size of one or more individual DSAs by using the corresponding SIT overrides (**CDSASZE**, **UDSASZE**, **SDSASZE**, **RDSASZE**, **ECDSASZE**, **EUDSASZE**, **ESDSASZE**, and **ERDSASZE**). For more information about these overrides, see The system initialization parameter descriptions in the *CICS System Definition Guide*. To determine the values to use, follow this process:
   a. Collect DFH0STAT output for information showing storage use by each DSA during the intervals.
   b. Review the CICS statistics for several days. The statistics provide information that you can use 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 results about each subpool are provided, including the DSA where it was allocated. If statistics are being gathered, end-of-day statistics only provide data since the last statistics collection.

CICS subpools

In each dynamic storage area, storage is arranged in subpools. The CICS subpools are dynamically acquired as needed, a page at a time, from within the applicable dynamic storage area.

Most CICS subpools are in 31-bit (above-the-line) or 64-bit (above-the-bar) storage. The subpools in 24-bit (below-the-line) storage must be monitored more carefully because of the limited space available.

Individual subpools can be static or dynamic. Some subpools contain static CICS storage, which cannot be tuned. All the subpools are rounded up to a multiple of 4 KB in storage size. Include this rounding factor in any subpool sizing, or evaluation of storage size changes after tuning or other changes.

The CICS domain subpools statistics contain useful information about the size and use of the dynamic storage area subpools. The following topics list the subpools in each dynamic storage area and their use. You can use this information to identify the possible causes of excessive usage in individual subpools.

CICS subpools in the CDSA

The subpools in the CICS dynamic storage area (CDSA) are listed, together with the use of each one.

Table 6. CICS subpools in the CDSA

Subpool name	Description
AP_TCA24	Storage for the TCA when the task data location option is set to BELOW.
DFHAPD24	A general subpool for 24-bit application domain storage.
DFHTDSDS	Storage for real transient data SDSCIs, each of which contains a DCB that resides in 24-bit storage (below 16 MB).
DHPDPOOL	Storage for DCBs for partitioned data sets used by document handler domain.
FC_DCB	Storage for the DCBs for BDAM files. Each file that is defined requires 104 bytes.
FCCBELOW	Storage for 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	A single 2 KB 24-bit (below 16 MB) stack segment. This is a dummy stack segment that is shared by all tasks. Tasks that need to use 24-bit stack storage obtain an extension stack segment from the subpool KESTK24E.
KESTK24E	4 KB 24-bit (below 16 MB) extension stack segments obtained by tasks that need to use 24-bit stack storage. CICS preallocates a reserve pool of 24-bit extension stack segments that tasks can use if no other storage is available in the subpool.
LD_JFCB	Storage for the job file control blocks for the loader domain.

Table 6. CICS subpools in the CDSA (continued)

Subpool name	Description
LDNRS	Storage for the CICS nucleus and macro tables that are RESIDENT. The CICS nucleus is approximately 192 KB and the size of the tables can be calculated. Programs are defined as EXECKEY (CICS) and link edited with RMODE(24) without the reentrant open.
LDNUC	Storage for the CICS nucleus and macro tables that are not RESIDENT. The CICS nucleus is approximately 192 KB and the size of the tables can be calculated. Programs are defined as EXECKEY (CICS) and link edited with RMODE(24) without the reentrant open.
SMCONTRL	Satisfies GETMAIN requests for control class storage.
SMSHARED	24-bit shared storage, for example RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
SMSHRC24	Used for many control blocks of SHARED_CICS24 class storage.
SMTP24	Storage for line and terminal I/O areas that cannot be located above 16 MB. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool can be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
SZSPFCAC	FEPI z/OS Communications Server ACB work areas.
TRUBELOW	Task-related user exit pool below 16 MB.
XMGEN24	General storage used by transaction manager.
ZCSETB24	Application control buffers below 16 MB.
ZCTCTUA	Storage for the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, 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. For more information about the terminal definition, see TERMINAL resources in Reference -> System definition.

CICS subpools in the SDSA

The subpools in the shared dynamic storage area (SDSA) are listed, together with the use of each one.

Table 7. CICS subpools in the SDSA

Subpool name	Description
APECA	Storage for the event control areas.
DFHAPU24	A general subpool for application domain storage below 16 MB.
LDPGM	Storage for dynamically loaded application programs (RMODE (24)). The expected size of this subpool can be predicted from previous releases, and by taking LDPGMRO into account. The subpool size can be reduced by using 31-bit programs. Not reentrant.
LDRES	Storage for resident application programs (RMODE (24). The expected size of this subpool can be predicted from previous releases, and by taking LDRESRO into account. The subpool size can be reduced by using 31-bit programs. Not reentrant.
OSCOBOL	Used for the allocation of the COBOL merged load list (MLL) control block and its extents. This subpool should never occupy more than its initial allocation of one page of storage.

Table 7. CICS subpools in the SDSA (continued)

Subpool name	Description
SMSHRU24	Used for many control blocks of SHARED_USER24 class storage.
ZCTCTUA	Storage for the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, 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. For more information about the terminal definition, see TERMINAL resources in Reference -> System definition.

CICS subpools in the RDSA

The subpools in the read-only dynamic storage area (RDSA) are listed, together with the use of each one.

Table 8. CICS subpools in the RDSA

Subpool name	Description
LDNRSRO	Storage for programs that are defined EXECKEY(CICS) that are RESIDENT, that were link edited REENTRANT and RMODE(24).
LDNUCRO	Storage for programs that are defined EXECKEY(CICS) that are not RESIDENT, that were link edited REENTRANT and RMODE(24).
LDPGMRO	Storage for programs that are defined EXECKEY(USER) that are not RESIDENT, that were link edited RMODE(24) and REENTRANT.
LDRESRO	Storage for programs that are defined EXECKEY(USER) and RESIDENT and were link edited REENTRANT and RMODE(24).

CICS subpools in the ECDSA

The subpools in the extended CICS dynamic storage area (ECDSA) are listed, together with the use of each one.

Table 9. CICS subpools in the ECDSA

Subpool name	Description
>LGJMC	Storage for the journal model resource entries for the log manager domain.
AITM_TAB	The autoinstall terminal model (AITM) table entry subpool (DFHAITDS).
AP_TCA31	Storage for the TCA when the task data location option is set to ANY.
AP_TXDEX	Storage for the application part of the TXD table.
APAID31	Storage for AIDs above the line.
APBMS	Storage used by BMS.
APCOMM31	Storage for COMMAREAs. The storage requirement depends on the size of COMMAREA specified and the number of concurrent users of the application.
APDWE	Storage for non-task deferred work elements.
APICE31	Storage for ICEs above the line.
APURD	Subpool containing unit of recovery descriptors (URDs) and nontask deferred work elements (DWEs).
ASYNCBUF	Buffers used by asynchronous operations in the sockets domain.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
BAGENRAL	A general-purpose subpool for the business application manager domain.
BAOFBUSG	Buffer storage used by the business application manager domain.
BAOFT_ST	Storage used by activities in the business application manager domain.
BR_BFBE	Storage for the bridge facility block extension.
BR_BFNB	Storage for the bridge facility name block.
BR_BMB	Storage for the bridge message block.
BR_BSB	Storage for bridge start blocks.
BRGENRAL	General-purpose subpool used by the bridge.
BRNSBLK	Storage used for the bridge numberspace.
BRNSFBLK	Storage used for bridge files.
BRPC	Storage used for bridge primary clients.
BRVS	Storage used for bridge virtual terminals.
BRVSCA	Storage used for bridge virtual screen character attributes.
BRVSXA	Storage used for bridge virtual screen extended attributes.
CCNV_BCE	Storage for character conversion buffer chain elements.
CCNV_CCE	Storage for character conversion chain elements.
CCNV_TRT	Storage for character conversion translation tables. These tables are addressed by the conversion chain elements.
CCNVG_AN	Storage for character conversion anchor blocks.
COLARAY	Storage for web control block array storage.
CQCQ_AN	Storage for console queue management anchor blocks.
CQCQ_CB	Storage for console queue management command input buffers.
DBCTL	Subpool that contains the TIE blocks for RMI use, when called 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 can be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DBDBG	Storage for DBCTL global blocks.
DCTE_EXT	Storage for all extrapartition queue definitions.
DCTE_IND	Storage for all indirect queue definitions.
DCTE_INT	Storage for all intrapartition queue definitions.
DCTE_REM	Storage for all remote queue definitions.
DDAPSESS	Storage for LDAP sessions state control blocks.
DDAPSRCH	Buffers for LDAP search results.
DDBROWSE	Storage for directory manager browse request tokens.
DDGENRAL	Storage for directory manager control blocks general information.
DDS_BFBE	Storage for directory manager directory elements for the BFBE table.
DDS_BFNB	Storage for directory manager directory elements for the BFNB table.
DDS_DCTE	Storage for directory manager directory elements for the DCTE table.
DDS_DHT1	Storage for directory manager directory elements for the DHT1 table.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
DDS_DHT2	Storage for directory manager directory elements for the DHT2 table.
DDS_DSN	Storage for directory manager directory elements for the DSN table.
DDS_D2AC	Storage for directory manager directory elements for the D2AC table.
DDS_D2CS	Storage for directory manager directory elements for the D2CS table.
DDS_D2EN	Storage for directory manager directory elements for the D2EN table.
DDS_D2TN	Storage for directory manager directory elements for the D2TN table.
DDS_D2TT	Storage for directory manager directory elements for the D2TT table.
DDS_ECCS	Storage for directory manager directory elements for the ECCS table.
DDS_ECEV	Storage for directory manager directory elements for the ECEV table.
DDS_ECSC	Storage for directory manager directory elements for the ECSC table.
DDS_EPAD	Storage for directory manager directory elements for the EPAD table.
DDS_FCT	Storage for directory manager directory elements for the FCT table.
DDS_ISIA	Storage for directory manager directory elements for the ISIA table.
DDS_ISIN	Storage for directory manager directory elements for the ISIN table.
DDS_JVMD	Storage for directory manager directory elements for the JVMD table.
DDS_MLRL	Storage for directory manager directory elements for the MLRL table.
DDS_MLXT	Storage for directory manager directory elements for the MLXT table.
DDS_MQII	Storage for directory manager directory elements for the MQII table.
DDS_MQIN	Storage for directory manager directory elements for the MQIN table.
DDS_NQRN	Storage for directory manager directory elements for the NQRN table.
DDS_PIPL	Storage for directory manager directory elements for the PIPL table.
DDS_PPT	Storage for directory manager directory elements for the PPT table.
DDS_PTPO	Storage for directory manager directory elements for the PTPO table.
DDS_PTST	Storage for directory manager directory elements for the PTST table.
DDS_PTT	Storage for directory manager directory elements for the PTT table.
DDS_REFE	Storage for directory manager directory elements for the REFE table.
DDS_RLBN	Storage for directory manager directory elements for the RLBN table.
DDS_RTXD	Storage for directory manager directory elements for the RTXD table.
DDS_SCAC	Storage for directory manager directory elements for the SCAC table.
DDS_SERV	Storage for directory manager directory elements for the SERV table.
DDS_SOCI	Storage for directory manager directory elements for the SOCI table.
DDS_SOSI	Storage for directory manager directory elements for the SOSI table.
DDS_TCL	Storage for directory manager directory elements for the TCL table.
DDS_TPNM	Storage for directory manager directory elements for the TPNM table.
DDS_TXD	Storage for directory manager directory elements for the TXD table.
DDS_USD1	Storage for directory manager directory elements for the USD1 table.
DDS_USD2	Storage for directory manager directory elements for the USD2 table.
DDS_USD3	Storage for directory manager directory elements for the USD3 table.
DDS_USD4	Storage for directory manager directory elements for the USD4 table.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
DDS_WBST	Storage for directory manager directory elements for the WBST table.
DDS_WBUR	Storage for directory manager directory elements for the WBUR table.
DDS_WSRD	Storage for directory manager directory elements for the WSRD table.
DDS_WURS	Storage for directory manager directory elements for the WURS table.
DDS_W2AT	Storage for directory manager directory elements for the W2AT table.
DDS_W2RL	Storage for directory manager directory elements for the W2RL table.
DFHAPDAN	A general subpool for application domain storage above 16 MB but below 2 GB.
DFHD2CSB	Storage for control blocks representing DB2 threads created by the CICS/DB2 adapter.
DFHD2ENT	Storage for control blocks representing DB2ENTRY definitions.
DFHD2PKG	Storage for control blocks representing DB2 PACKAGESET definitions.
DFHD2TRN	Storage for control blocks representing DB2TRAN definitions.
DFHECCD	Storage for event capture data.
DFHECCS	Storage for event capture specification blocks.
DFHECDQE	Storage for event capture deferred filter queue elements.
DFHECEVB	Storage for event capture event binding blocks.
DFHECFP	Storage for event capture event filter predicate blocks.
DFHECSC	Storage for event capture system event calls.
DFHECSF	Storage for event capture system filter predicates.
DFHEPAC	Storage for event capture event adapter configuration data.
DFHMPGEN	Used for allocations of Managed Platform anchor block (MPA) and failed adapter (MPPFA) control blocks.
DFHMPMOD	Used for allocations of Managed Platform model (MPMOD) control blocks.
DFHMPPPB	Used for allocations of Managed Platform policy (MPPPB) control blocks.
DFHTDG31	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	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	Storage for the transient data wait elements.
DHCACHE	Storage for cached copies of document templates.
DHDBB	Storage for document bookmark blocks.
DHDCR	Storage for document control records.
DHDDB	Storage for document data.
DHDOA	Storage for document anchor blocks.
DHFSPATH	Storage for HFS path template extensions.
DHGENRAL	The general purpose subpool for the document manager domain.
DHSTB	Storage for document symbol tables.
DHTLPOOL	Storage for document handler template descriptors.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
DLI	Subpool that contains the TIE blocks for RMI use, when called 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 present only when EXEC DL/I is used. It can be tuned by limiting DBCTL threads or using maximum tasks (MXT) or transaction classes.
DMSUBPOL	The domain manager subpool for general usage.
DP_GENRL	Storage for the control blocks for the DP domain.
DPLA	Storage for the anchor blocks for instore linked lists of debugging profiles.
DPLE	Storage for the elements in the instore linked lists of debugging profiles.
DPLP	Storage for the elements in the debug profile that is used for pattern matching.
DPTA	Storage for transaction instance state data that is required by the DP domain.
DS_STIMR	Storage for dispatcher domain STIMER tokens.
DS_TCB	Storage for dispatcher domain TCBs.
DS_VAR	The dispatcher domain variable length subpool.
DSBROWSE	Storage for dispatcher browse request tokens.
EC_GENRL	Storage for the control blocks for the EC domain.
EJMI	The enterprise bean method information.
EJOSGENS	The enterprise bean general subpool.
EJOSTSKS	The enterprise bean task subpool.
EJSPBFBC	Storage for web browser control blocks for enterprise beans.
EJSPBVIC	Storage for enterprise bean control blocks.
EJSPCFBC	Storage for web browser control blocks for CorbaServers.
EJSPCFIC	Storage for control blocks for CorbaServers.
EJSPCOMM	Storage for anchor blocks for enterprise beans.
EJSPDFBC	Storage for web browser control blocks for deployed JAR files.
EJSPDFIC	Storage for control blocks for deployed JAR files.
EJSPGVNC	Storage for persistent storage for enterprise beans.
EJSPTVNC	Storage for transaction-related storage for enterprise beans.
EJSTGENS	Storage for control blocks for enterprise bean statistics.
EMBRB	Storage for event manager browse blocks.
EMEVA	Storage for the event manager event pool anchor.
EMEVB	Storage for event manager event blocks.
EMGENRAL	General-purpose subpool for event manager domain.
EP_GENRL	Storage for the control blocks for the EP domain.
EPADA	Storage for event processing adapter management.
EPADI	Storage for EP adapter name in EP adapter set.
EPADT	Storage for event processing adapter set management.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
FC_ABOVE	Storage for real VSWA and data buffers for prereads. Each VSWA requires 120 bytes of storage. The maximum number of data buffers for prereads is given by: (number of strings) x (maximum record length) x (number of files)
FC_ACB	Storage for ACBs for VSAM files. Each VSAM file has one ACB, of 80 bytes.
FC_BDAM	Storage for BDAM file control blocks. Each BDAM file requires 96 bytes of storage.
FC_DSNAM	Storage for data set name blocks. Each file requires a data set name block, which uses 120 bytes of storage.
FC_FCPE	Storage for file control pool elements.
FC_FCPW	Storage for file control CFDT pool wait elements.
FC_FCUP	Storage for the file control CFDT unit of work pool block.
FC_FLAB	Storage for file control lasting access blocks.
FC_FLLB	Storage for file control lock locator blocks.
FC_FRAB	Storage for file request anchor blocks (FRABs). Each transaction that has issued a file control request has one FRAB. The FRAB is retained until the end of the task. There is a free chain of FRABs not currently in use.
FC_FRTE	Storage for 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 meets these conditions: • It has not yet stopped its VSAM thread. For example, a browse that has not yet issued an ENDBR. • It has updated a recoverable file and a sync point has not yet occurred. • It is holding READ-SET storage that must be freed in future. There is a free chain of FRTEs not currently in use.
FC_RPL	Storage for file control request parameter lists.
FC_SHRCT	Storage for file control SHRCTL blocks. There are eight of these blocks and each describes a VSAM LSR pool.
FC_VSAM	Storage for the file control table (FCT) entries for VSAM files.
FCB_256	File control buffers of length 256 bytes. These buffers are used by file control requests that are made against files with a maximum record length less than or equal to 256 bytes.
FCB_512	File control buffers of length 512 bytes. These buffers are used by file control requests that are made against files with a maximum record length between 256 + 1 bytes and 512 bytes.
FCB_1K	File control buffers of length 1 KB. These buffers are used by file control requests that are made against files with a maximum record length between 512 + 1 bytes and 1 KB.
FCB_2K	File control buffers of length 2 KB. These buffers are used by file control requests that are made against files with a maximum record length between 1 KB + 1 byte and 2 KB.
FCB_4K	File control buffers of length 4 KB. These buffers are used by file control requests that are made against files with a maximum record length between 2 KB + 1 byte and 4 KB.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
FCB_8K	File control buffers of length 8 KB. These buffers are used by file control requests that are made against files with a maximum record length between 4 KB + 1 byte and 8 KB.
FCB_16K	File control buffers of length 16 KB. These buffers are used by file control requests that are made against files with a maximum record length between 8KB + 1 byte and 16 KB.
FCB_32K	File control buffers of length 32 KB. These buffers are used by file control requests that are made against files with a maximum record length between 16 KB + 1 byte and 32 KB.
FCB_64K	File control buffers of length 64 KB. These buffers are used by file control requests that are made against files with a maximum record length between 32 KB + 1 byte and 64 KB.
FCB_128K	File control buffers of length 128 KB. These buffers are used by file control requests that are made against files with a maximum record length between 64 KB + 1 byte and 128 KB.
FCB_256K	File control buffers of length 256 KB. These buffers are used by file control requests that are made against files with a maximum record length between 128 KB + 1 byte and 256 KB.
FCB_512K	File control buffers of length 512 KB. These buffers are used by file control requests that are made against files with a maximum record length between 256 KB + 1 byte and 512 KB.
FCB_1M	File control buffers of length 1MB. These buffers are used by file control requests that are made against files with a maximum record length between 512 KB + 1 byte and 1 MB.
FCB_2M	File control buffers of length 2 MB. These buffers are used by file control requests that are made against files with a maximum record length between 1 MB + 1 byte and 2 MB.
FCB_4M	File control buffers of length 4 MB. These buffers are used by file control requests that are made against files with a maximum record length between 2 MB + 1 byte and 4 MB.
FCB_8M	File control buffers of length 8 MB. These buffers are used by file control requests that are made against files with a maximum record length between 4 MB + 1 byte and 8 MB.
FCB_16M	File control buffers of length 16 KB. These buffers are used by file control requests that are made against files with a maximum record length between 8 MB + 1 byte and 16 MB.
FCSTATIC	File control static storage.
ICUS	Storage for internal control element (ICE) secure extensions.
IE_GENRL	Storage for the control blocks for the IE domain.
IECCB	Storage for the conversation control blocks in the IE domain.
IECSB	Storage for the client state blocks in the IE domain.
IFGLUWID	The VSAM IFGLUWID area.
IIGENRAL	The IIOP domain general subpool.
IIMBR	The IIOP domain request model browse block.
IIMDB	The IIOP domain request model block.
IS_GENRL	Storage for the control blocks for the IS domain.
ISAQ	Storage for IS allocate queue elements.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
ISCB	Storage for IS control blocks, used to record installed instances of IPCONNs.
ISQA	Storage for IS queue attach control blocks.
ISRD	Storage for IS remote delete requests.
ISSB	Storage for the IS session blocks, each of which is associated with an ISCB subpool.
ISSS	Storage for IS session sets.
KEANCHOR	Storage Manager domain anchors.
KESTK31	28 KB 31-bit (above the line) stack segments. There is one per MXT plus one for every dynamic system task that is running.
KESTK31E	8 KB 31-bit (above the line) extension stack segments. There is at least one for every ten tasks specified in the MXT limit.
KETASK	Storage for kernel task entries.
LD_CDE	Storage for loader domain dummy CDEs.
LD_CNTRL	Storage for loader domain general control information.
LD_LDBE	Storage for LDBE (loader browse element) control blocks for the loader domain.
LD_LDWE	Storage for LDWE (loader suspend work element) control blocks for the loader domain.
LD_PLIBE	Storage for program library element storage for the loader domain.
LDENRS	Storage for the extended CICS nucleus, and 31-bit macro tables that are RESIDENT. The extended CICS nucleus is approximately 50 KB. Programs are defined with EXECKEY(CICS) and link edited RMODE(ANY) without the REENTRANT option.
LDENUC	Storage for the extended CICS nucleus and 31-bit macro tables that are not RESIDENT. The extended CICS nucleus is approximately 50 KB. Programs are defined with EXECKEY(CICS) and link edited RMODE(ANY) without the REENTRANT option.
LGBD	Storage for log stream name, journal name, and journal model browse tokens for the log manager domain.
LGGD	Storage for explicitly opened general logs for the log manager domain.
LGGENRAL	The general-purpose subpool for the log manager domain.
LGJI	Storage for journal name entries for the log manager domain.
LGSD	Storage for log stream data entries for the log manager domain.
LGUOW	Storage for unit-of-work data entries for the log manager domain.
LI_PLB	Storage for the language interface program language block. One is allocated for each program when control is first passed to it.
L2GENRAL	The log manager domain general subpool.
L2OFL2BL	Storage for logger block entries for the log manager domain.
L2OFL2BS	Storage for browseable stream objects for the log manager domain logger.
L2OFL2CH	Storage for chain objects for the log manager domain logger.
L2OFL2SR	Storage for stream objects for the log manager domain logger.
MDTTABLE	The MDT field attribute table for BMS maps sent through the CICS web interface.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
ML_GENRL	General storage for the ML domain.
MN_CNTRL	Storage for monitoring control blocks general information.
MN_TIMAS	Storage for monitoring control blocks identity monitoring area.
MN_TMAS	Storage for monitoring control blocks transaction monitoring area.
MN_TRMAS	Storage for monitoring control blocks resource monitoring area.
MQM	WebSphere MQ communication storage.
MRO_QUEU	Used by the MRO work queue manager.
MROWORKE	Used by the MRO work queue manager elements.
NQEAS	Storage for NQ domain queue element areas.
NQGENRAL	A general subpool used by NQ domain.
NQPOOL	Storage for NQ domain enqueue pools.
NQRNAMES	Storage for NQRN directory entries.
OTGENRAL	The general subpool used by the OT domain.
OTISINST	Storage for inflight state of OTS transactions.
OVERLAPD	Storage for overlap field merging.
PGCHCB	Storage for channel control blocks. This storage contains header information that describes a channel.
PGCPCB	Storage for the channel container pool control block. This storage contains header information that describes sets of containers.
PGCPCBCH	Storage for the chained container pool control block.
PGCRBB	Storage for browses of channel containers.
PGCRCB	Storage for channel container control blocks. This storage contains the header information for each container.
PGCSCB	Storage for channel container segments.
PGGENRAL	Storage for general purpose program manager domain subpools.
PGHMRSA	Storage for program handle manager cobol register save areas.
PGHTB	Storage for the program manager handle table block.
PGJVMCL	Storage for JVM class names.
PGLLE	Storage for program manager load list elements.
PGPGWE	Storage for program manager wait elements.
PGPPTE	Storage for program manager program definitions.
PGPTA	Storage for program manager transaction-related information.
PI_GENRL	General storage for the pipeline manager (PI) domain.
PI_POLCY	Currently not used.
PI_PRSER	Currently not used.
PINODEBL	Storage for pipeline objects.
PIPEINST	Storage for pipeline objects.
PITKDAT	Storage for pipeline token data for context token.
PITKPOOL	Storage for pipeline tokens.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
PITXMAST	Storage for Web Services Atomic Transaction (WS-AT) master control block or PI domain transaction control block.
PR_TABLE	Storage for PTEs from the PRT.
PTTWSB	General storage for pool tokens.
RCLELEM	Storage for the web row-column element list storage.
RCTABLE	Web table storage.
RLGENRAL	The resource lifecycle general subpool.
RMGENRAL	The recovery manager general subpool.
RMOFRMLK	Storage for recovery manager link objects.
RMOFRMUW	Storage for recovery manager unit-of-work objects.
ROWARAY	Web row array storage.
RS_FILEL	Region status (RS) domain file list storage.
RS_GENRL	Storage for the control blocks for the RS domain.
RUNTRAN	A transaction manager subpool for run transaction.
RUTKPOOL	A subpool for reusable token class.
RXGENRAL	A general subpool for RX domain.
RZGENRAL	A general subpool for request streams domain.
RZOFRSNR	Storage for request streams notification requests.
RZOFRSRG	Storage for request streams registration objects.
RZOFRZRS	Storage for request streams objects.
RZOFRZTR	Storage for request stream transports.
SHGENRAL	The general subpool for scheduler services domain.
SHOFSHRE	Storage for scheduler services request objects.
SJGENRAL	The general subpool for SJVM domain.
SJJ8TCB	Storage for TCBs in the SJVM domain.
SMSHRC31	Storage for many control blocks of the SHARED_CICS31 class.
SMTP	Line and terminal I/O areas. The storage requirements depend on the amount of terminal and line traffic in the system. The subpool can be tuned by reducing the RAPOOL, RAMAX, TIOAL size, and number of MRO sessions.
SOCKET	Storage for socket objects.
SOCKPOOL	Socket pool storage.
SOCKSSL	Storage for the SSL data related to a socket.
SOGENRAL	The sockets domain general subpool.
SOLTE	Storage for socket domain listener terminal entries.
SOSTE	Storage for socket domain socket terminal entries.
SOTBR	Storage for socket domain TCPIPSERVICE browse blocks.
SOTDB	Storage for socket domain TCPIPSERVICE blocks.
SOTKPOOL	Storage for socket domain socket tokens.
STSUBPOL	A statistics domain manager subpool.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
SZSPFCCD	The FEPI connection control subpool.
SZSPFCCM	The FEPI common area subpool.
SZSPFCCV	The FEPI conversation control subpool.
SZSPFCDS	The FEPI device support subpool.
SZSPFCNB	The FEPI node initialization block subpool.
SZSPFCND	The FEPI node definition subpool.
SZSPFCPD	The FEPI pool descriptor subpool.
SZSPFCPS	The FEPI property descriptor subpool.
SZSPFCRP	The FEPI request parameter list subpool.
SZSPFCRQ	The FEPI requests subpool.
SZSPFCSR	The FEPI surrogate subpool.
SZSPFCTD	The FEPI target descriptor subpool.
SZSPFCWE	The FEPI work element subpool.
SZSPVUDA	The FEPI data areas subpool.
TA_GENRL	Currently not used.
TASKASOC	Storage for sockets domain task association objects.
TD_TDCUB	Storage for all the transient data CI update control blocks.
TD_TDQUB	Storage for all the transient data queue update control blocks.
TD_TDUA	Storage for all the transient data UOW anchor control blocks.
TFUS	Storage for TCTTE secure extensions.
TIA_POOL	The timer domain anchor subpool.
TIQCPOOL	The timer domain quick cell subpool.
TSBRB	Storage for temporary storage (TS) browse blocks.
TSBUFFRS	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. The amount depends on the number of buffers and strings and the control interval size defined for the temporary storage data set.
TSICDATA	Storage for TS interval control elements.
TSMBR	Storage for temporary storage browse blocks.
TSMDB	Storage for temporary storage model blocks.
TSQAB	Storage for TS queue anchor blocks.
TSQOB	Storage for TS queue ownership blocks.
TSQUB	Storage for TS queue update blocks.
TSTSS	Storage for TS section descriptors.
TSTSX	Storage for TS auxiliary item descriptors.
TSW	Storage for TS wait queue elements.
UE_EPBPL	The subpool for the user exit program block (EPB).

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
USIDTBL	Storage for the attach security userid table entries (LUITs). See "ISC/IRC attach time entry statistics" on page 559 for more information.
WBGENRAL	The general subpool for CICS web support.
WBOUTBND	Storage for outbound HTTP buffers.
WBPATHN1	Storage for path node elements used for URI map storage for short path names.
WBPATHN2	Storage for path node elements used for URI map storage for long path names.
WBPATHUR	Storage used for URI map storage for URI path names.
WBRQB	Storage for web request objects.
WBS	Storage for inbound web session blocks used for the IPIC protocol.
WBURIMAP	Storage for URI mapping elements.
WBURIXT1	Storage for URI mapping element extensions (short).
WBURIXT2	Storage for URI mapping element extensions (long).
WBWRBR	Storage for web request browse blocks.
WBVHOST	Storage for URI virtual host elements.
WEB_STA	Web state-related storage.
WEBELEM	Storage for web output element lists.
WEBHTML	Storage for web HTML buffers.
WEBINB	Storage for web domain storage for incoming data.
WEB327B	Web domain 3270 buffer storage.
W2ATOMSE	Storage for Web 2.0 atom service elements.
W2ATOMX1	Storage for Web 2.0 atom service extensions.
W2ATOMX2	Storage for Web 2.0 atom service extensions.
W2GENRAL	The general-purpose subpool for the Web 2.0 domain.
XMGENRAL	The general-purpose subpool for the transaction manager.
XMTCLASS	Storage for the transaction manager tranclass definition.
XMTRANSN	Storage for transaction manager transactions; one for every transaction in the system.
XMTXDINS	The transaction manager transaction definition.
XMTXDSTA	The transaction manager transaction definition.
XMTXDTPN	The transaction manager transaction definition TPNAME storage.
ZC2RPL	Storage for the duplicate RPLs for active tasks. Each active task associated with a z/OS Communications Server terminal requires 304 bytes.
ZCBIMG	Storage for BIND images.
ZCBMSEXT	Storage for the BMS extensions for terminals. Subpool storage requirements are 48 bytes for each terminal, surrogate, ISC session, and console.
ZCBUF	Storage for the non-LU 6.2 buffer list.
ZCCCE	Storage for the console control elements. Each console requires 48 bytes.
ZCGENERL	The general-purpose subpool for terminal control.
ZCLUCBUF	Storage for the LU 6.2 SEND and RECEIVE buffer list.

Table 9. CICS subpools in the ECDSA (continued)

Subpool name	Description
ZCLUCEXT	Storage for the LU 6.2 extensions. The storage requirement is 224 bytes for each LU 6.2 session.
ZCNIBD	Storage for the NIB descriptors. Each terminal, surrogate, ISC session, and system definition requires 96 bytes of storage.
ZCNIBISC	Storage for the expanded NIB and response during OPNDST and CLSDST for ISC. Each concurrent logon and logoff requires 448 bytes of storage. The maximum number of concurrent requests is limited by the number of sessions. The storage can be tuned by reducing the number of sessions.
ZCNIBTRM	Storage for the expanded NIB during OPNDST and CLSDST for terminals. Each concurrent logon and logoff requires 192 bytes of storage. The maximum number of concurrent requests is limited by the number of terminals. The storage can be tuned by reducing the number of terminals.
ZCRAIA	Storage for the RECEIVE ANY I/O areas.
ZCRPL	Storage for the RPLs for active tasks. Each active task associated with a z/OS Communications Server terminal requires 152 bytes.
ZCSETB	Storage for application control buffers above 16 MB but below 2 GB.
ZCSKEL	Storage for the remote terminal entries. Each remote terminal definition requires 32 bytes of storage.
ZCSNEX	Storage for the TCTTE sign-on extensions. The storage requirement is 48 bytes for each terminal, surrogate, session, and console.
ZCTCME	Storage for the mode entries. Each mode entry requires 128 bytes of storage.
ZCTCSE	Storage for the system entries. Each system entry requires 192 bytes of storage.
ZCTCTTEL	Storage for the large terminal entries. 504 bytes of storage are required for every terminal, surrogate model, and ISC session defined.
ZCTCTTEM	Storage for the medium terminal entries. 400 bytes of storage are required for every IRC batch terminal.
ZCTCTTES	Storage for the small terminal entries. 368 bytes of storage are required for every MRO session and console.
ZCTPEXT	The TPE extension.
ZCTREST	The terminal control transaction restart subpool.
ZCTCTUA	Storage for the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, 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. For more information about the terminal definition, see TERMINAL resources in Reference -> System definition.

CICS subpools in the ESDSA

The subpools in the extended shared dynamic storage area (ESDSA) are listed, together with the use of each one.

Table 10. CICS subpools in the ESDSA

Subpool name	Description
DFHAPUAN	A general subpool for application domain storage above 16 MB but below 2 GB.
IE_BUFF	The IE domain buffers that are used when processing inbound and outbound messages.
IIBUFFER	The IIOP domain buffer subpool.
IS_BUFF	Storage for the IS buffers that are used to hold the message data for an IS session block.
LDEPGM	Storage for extended (31-bit) dynamically-loaded application programs and programs defined EXECKEY(USER).
LDERES	Storage for extended (31-bit) resident application programs.
SJSCCHS	Storage for the Java Virtual Machine domain (SJ domain) class cache.
SJSJPTE	Storage for the SJ domain profile table entries.
SJSJTCB	Storage for the SJ domain TCB usage.
SJUSERKY	SJ domain user key storage.
SMSHRU31	Used for many control blocks of SHARED_USER31 class storage, RMI global work areas, EDF blocks for the life of the transaction being monitored, and other control blocks.
TGODR	Storage for the transaction group origin data record.
WEBINB	Inbound Web 3270 buffer storage.
ZCTCTUA	Storage for the TCTTE user area. It can be located in one of the following DSAs: SDSA, ECDSA, CDSA, 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. For more information about the terminal definition, see TERMINAL resources in Reference -> System definition.

CICS subpools in the ERDSA

The subpools in the extended read-only dynamic storage area (ERDSA) are listed, together with the use of each one.

Table 11. CICS subpools in the ERDSA

Subpool name	Description
LDENRSRO	Storage for the extended CICS nucleus and 31-bit macro tables that are RESIDENT. The extended CICS nucleus is approximately 1850 KB. The contents of this subpool must be linked reentrant.
LDENUCRO	Storage for the extended CICS nucleus and 31-bit macro tables that are not RESIDENT. The extended CICS nucleus is approximately 1850 KB. The contents of this subpool must be linked reentrant.
LDEPGMRO	Storage for extended (31-bit) dynamically loaded application programs. The contents of this subpool must be linked reentrant.
LDERESRO	Storage for extended (31-bit) resident application programs. The contents of this subpool must be linked reentrant.

CICS subpools in the ETDSA

The subpools in the extended trusted dynamic storage area (ETDSA) are listed, together with the use of each one.

Table 12. CICS subpools in the ETDSA

Subpool name	Description
USGENRAL	The general purpose subpool for the user domain.
USRTMQUE	Storage for queue elements for users waiting for USRDELAY. Each queue element is 16 bytes.
USUDB	Storage for user data blocks. The storage requirement is 128 bytes for each unique user.
USXDPOOL	Storage for user domain transaction-related data. Each running transaction requires 32 bytes.
XSGENRAL	The general purpose subpool for the security domain.
XSXMPOOL	Storage for security domain transaction-related data. Each running transaction requires 56 bytes.

CICS subpools in the GCDSA

The subpools in the above-the-bar CICS dynamic storage area (GCDSA) are listed, together with the use of each one.

Table 13. CICS subpools in the GCDSA

Subpool name	Description
CPSM_64	Storage for CPSM API result sets in a System Management Single Server (SMSS) environment.
CQCQ_TR	Storage for the console queue processing trace table.
CQCQ_XT	Storage for the console queue transaction entry table.
DFHAPD64	A general subpool for 64-bit application domain storage.
DFHMPMDR	Used for allocations of Managed Platform model rules (MPMODR) control blocks.
DFHMPPMB	Used for allocations of Managed Platform policy modifier (MPPMB) control blocks.
DFHMPPRB	Used for allocations of Managed Platform policy rule (MPPRB) control blocks.
DFHMPTAS	Used for allocations of Managed Platform task lifetime storage (MPTAS) control blocks.
EP_64	Storage for control blocks for items in event capture queues, used in CICS event processing.
LD_APES	Active Program Element (APE) control blocks for the loader domain.
LD_CPES	Current Program Element (CPE) control blocks for the loader domain.
LD_CSECT	CSECT list storage for the loader domain.
ML64GNRL	Buffers for input and output for the z/OS XML System Services (XMLSS) parser.
MN_ADCS	Storage for association data control blocks.
PGCSDB	Storage for channel container segments, including segment headers.
PGPPTE64	Storage for application context data control blocks.

Table 13. CICS subpools in the GCDSA (continued)

Subpool name	Description
SMSHRC64	Used for many control blocks of SHARED_CICS64 class storage.
TSDTN	Temporary storage (TS) digital tree nodes.
TSMAIN	Storage for main temporary storage.
TSMN0064	Fixed length elements for main temporary storage items that have lengths, including the header, less than or equal to 64 bytes. The header length is 8 bytes.
TSMN0128	128-byte fixed length elements for main temporary storage items.
TSMN0192	192-byte fixed length elements for main temporary storage items.
TSMN0256	256-byte fixed length elements for main temporary storage items.
TSMN0320	320-byte fixed length elements for main temporary storage items.
TSMN0384	384-byte fixed length elements for main temporary storage items.
TSMN0448	448-byte fixed length elements for main temporary storage items.
TSMN0512	512-byte fixed length elements for main temporary storage items.
TSMN0576	576-byte fixed length elements for main temporary storage items.
TSMN0640	640-byte fixed length elements for main temporary storage items.
TSMN0704	704-byte fixed length elements for main temporary storage items.
TSMN0768	768-byte fixed length elements for main temporary storage items.
TSMN0832	832-byte fixed length elements for main temporary storage items.
TSMN0896	896-byte fixed length elements for main temporary storage items.
TSMN0960	960-byte fixed length elements for main temporary storage items.
TSMN1024	1024-byte fixed length elements for main temporary storage items.
TSQUEUE	TS queue descriptors.
TSTSI	TS item descriptors.
WU_64	Storage for CMCI retained results and metadata.
XMGEN64	A general subpool for 64-bit storage.

CICS subpools in the GSDSA

The subpools in the above-the-bar shared dynamic storage area (GSDSA) are listed, together with the use of each one.

Table 14. CICS subpools in the GSDSA

Subpool name	Description
DFHAPU64	A general subpool for application domain storage above the bar.
SMSHRU64	Used for many control blocks of SHARED_USER64 class storage.

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. Most of this storage is allocated from the CICS DSAs. A small amount of this storage is allocated from MVS storage.

The kernel recognizes two types of task: static tasks and dynamic tasks. The kernel storage for static tasks is preallocated and is used for tasks controlled by the MXT mechanism. The storage for dynamic tasks is not preallocated 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 preallocated, the kernel might need to use a GETMAIN command to obtain the storage required to attach a dynamic task when the task is attached.

The number of static tasks depends on the current MXT value. There are MXT+1 static tasks. The storage for static tasks is always obtained by GETMAIN 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 eight dynamic tasks. This storage is obtained by GETMAIN 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 that CICS allocates during task initialization for a single task is the same for a static or dynamic task, as follows:

- A 1576-byte kernel task entry
- A 28K 31-bit stack

The allocated storage is all above the 16 MB line. CICS no longer allocates a 24-bit stack (below the line) for each task during task initialization.

In addition to the storage allocated at task initialization, the kernel also allocates pools of extension stack segments both above and below the 16 MB line.

- The size of each 31-bit extension stack segment (above the line) is 8 KB. Any task can use these extension stack segments if it overflows the 31-bit stack storage allocated to it. CICS preallocates a pool containing a number of 31-bit extension stack segments that is determined by dividing the current MXT value by 10.
- The size of each 24-bit extension stack segment (below the line) is 4 KB. Tasks obtain these extension stack segments whenever they require 24-bit stack storage. CICS preallocates a reserve pool of 24-bit extension stack segments that tasks can use if no other 24-bit stack storage is available.

When the kernel obtains storage using GETMAIN from the CICS DSAs, the following subpools are used:

KESTK24E in the CDSA
: 4 KB extension stack segments, 24-bit

KESTK31 in the ECDSA
: 28 KB stack segments, 31-bit

KESTK31E in the ECDSA
: 8 KB extension stack segments, 31-bit

KETASK in the ECDSA
: 1576-byte kernel task entries

64-bit MVS storage

64-bit MVS storage is available to the operating system to perform region-related services.

For information about 64-bit (above-the-bar) storage in an address space, see Using the 64-bit Address Space in the z/OS MVS Programming: Extended Addressability Guide.

If you run Java programs in a region, CICS uses the 64-bit JVM on z/OS. 64-bit MVS storage is allocated to each JVM that runs under the control of CICS.

For other CICS facilities that use 64-bit MVS storage, see "CICS facilities that use 64-bit storage" on page 101.

MVS storage below 2 GB

MVS storage below 2 GB is available to the 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. These requests are met from MVS storage below 2 GB.

MVS storage is the amount of storage that remains after the dynamic storage areas and other CICS storage requirements are met. The size of MVS storage below 2 GB depends on MVS GETMAIN requirements during the execution of CICS. Opening files is the major contributor to usage of this area.

MVS storage below 2 GB is used to contain the following items:
- Control blocks and data areas that are required to open data sets, or for other operating system functions
- Program modules for the access method routines that are not already resident in the link pack area (LPA)
- Shared routines for the COBOL and PL/I programs

There are four major elements of virtual storage in MVS storage below 2 GB. Each storage area below 16 MB is duplicated above 16 MB.
- The common area below 16 MB
- The private area below 16 MB
- The extended common area above 16 MB
- The extended private area above 16 MB

Storage when CICS uses other products

The VSAM buffers and most of the VSAM file control blocks reside above 16 MB. The VSAM buffers might be for CICS data sets defined as using local shared resources (LSR) or nonshared resources (NSR). The VSAM LSR pool is built dynamically above 16 MB 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 16 MB for records that are included in the table, and for the structures that allow them to be accessed.

Queued sequential access method (QSAM) files require some storage in this area. Transient data uses a separate buffer pool above 16 MB for each type of transient data queue. Storage is obtained from the buffer pool for transient data queue resources as they are installed. Transient data also uses a buffer pool above 16 MB where sections of extrapartition transient data queue definitions 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 CICS uses DB2, MVS storage is allocated for each DB2 thread.

MVS storage limits

The physical placement of the MVS storage below 2 GB can be anywhere in the region, and might sometimes be above the CICS region. The region might expand into this MVS storage area, above the region, up to the IEALIMIT set by the installation or up to the default value. For more information about IEALIMIT, see *z/OS MVS Installation Exits*. This expansion occurs when operating system GETMAIN requests are issued, the MVS storage in the region is exhausted, and the requests are met from the MVS storage area above the region.

When both the MVS storage areas below 2 GB are exhausted, the GETMAIN request fails, causing abends or a bad return code if it is a conditional request.

The amount of MVS storage below 2 GB must be enough to satisfy the requests for storage during the entire execution of the CICS region. You must use caution; you never want to run out of MVS storage, but you also do not want to allocate too much.

The size of MVS storage below 2 GB is the storage that 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. 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 you must allocate additional storage to compensate for this.

The MVS common area

The MVS common area contains a number of nucleus, queue, link pack, common service, and storage areas.

The following areas comprise the MVS common area:
- Nucleus and extended nucleus
- System queue area (SQA and ESQA)
- Link pack areas (PLPA, MLPA, and CLPA)
- Common service areas (CSA and ECSA)
- Prefixed storage area (PSA).

All these elements of the common area, except the PSA, are duplicated above 16 MB.

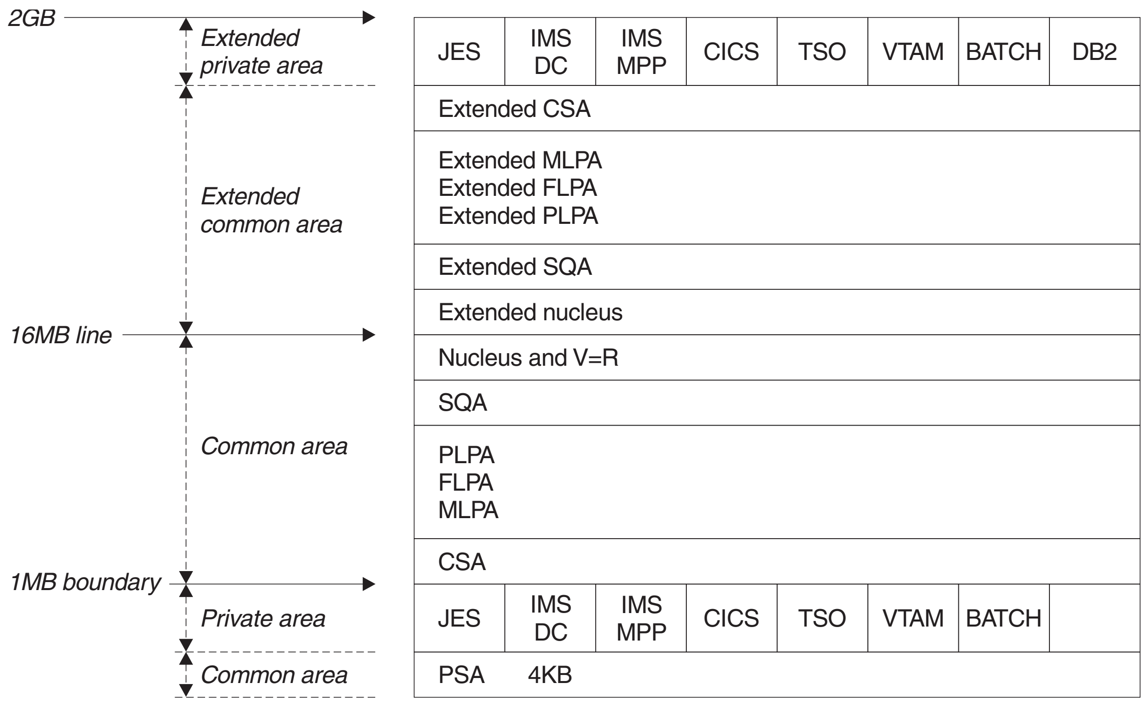


Figure 17. Virtual storage map

MVS nucleus and MVS extended nucleus:

The MVS nucleus and MVS extended nucleus is a static area that contains the nucleus load module and extension to the nucleus. Although its size is variable depending on the configuration of an installation, it cannot change without a re-IPL of MVS.

The nucleus area below 16 MB does not include page frame table entries, and the size of the nucleus area is rounded up to a 4 KB boundary. In addition, the nucleus area is positioned at the top of the 16 MB map while the extended nucleus is positioned just above 16 MB.

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 64 KB boundary, though SQA may expand into the common system area (CSA) in increments of 4 KB.

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 16 MB 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 that are shared by the system.

The link pack area (LPA) can provide the following:
- Economy of real storage by sharing one copy of the modules
- Protection: LPA code cannot be overwritten, even by key 0 programs
- Reduced path length, because modules can be branched to.

It has been established that a 2 MB 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 might be of no benefit to CICS. There might be SORT, COBOL, ISPF, and other modules that are benefiting batch and TSO users. You must evaluate whether the benefits you obtain are worth the virtual storage that they use. If modules are removed, check whether you need to increase the size of the regions they run in 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 can be changed at IPL without requiring the create link pack area (CLPA) option at IPL to change modules in the PLPA.

Common service area (CSA) and extended common service area (ECSA):

The CSA and ECSA contain pageable system data areas that are addressable by all active virtual storage address spaces.

These service areas contain, for example, buffers or executable modules for IMS, ACF/SNA, and JES3. CSA and ECSA also contain control blocks that are used to define subsystems and provide working storage for areas such as TSO input/output control (TIOC), event notification facility (ENF), and message processing facility (MPF). When system configuration and activity increases, the storage requirements also increase.

CICS uses the ECSA for multiregion operation (MRO) to store control blocks only and not for data transfer. If cross-memory facilities are used, the ECSA usage is limited to the following amounts:
- 40 bytes per session if IRC (interregion communication) is open, irrespective of whether the resource is acquired and inservice, or released
- 4 KB per address space participating in MRO

In addition, the amount of storage used by CICS MRO for interregion buffers is detailed in the DFHIR3794 message issued to the CSMT destination at termination.

CICS also uses ECSA for IMS and shared data tables.

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 optimal. Unlike the SQA, if CSA is depleted there is no scope for expansion and a re-IPL might 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 KB to 4 KB as seen by that particular processor.

MVS treats this as a separate area; there is no PSA in the extended common area.

Private area and extended private area

The portion of the user private area in each virtual address space that is available to the user's application program is referred to as its *region*. Except for the 16 KB system region area, each storage area in the private area has a counterpart in the extended private area.

The private area contains the following areas:

- A local system queue area (LSQA)
- A scheduler work area (SWA)
- Subpools 229 and 230 (the requestor protect key area)
- A 16 KB system region area (used by the initiator)
- A private user region for running programs and storing data.

See the virtual storage map for MVS in Figure 17 on page 132.

The private area user region can be any size up to the size of the entire private area (from the top end of the prefixed storage area (PSA) to the beginning, or bottom end, of the common service area (CSA)) **minus** the size of LSQA, SWA, subpools 229 and 230, and the system region: for example, 220 KB. It is recommended that the region is 420 KB less to allow for recovery termination management (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.

High private area

The area at the high end of the address space is not specifically used by CICS, but contains information and control blocks that the operating system needs to support the region and its requirements.

The high private area consists of four areas:

- LSQA
- SWA
- Subpool 229
- Subpool 230

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. A second SYS=nnnnK is issued, which refers to the high private area above 16 MB. This information is also reported in the sample statistics program, DFH0STAT.

You cannot reduce the size of this area, except possibly subpool 229. This subpool is where the z/OS Communications Server stores inbound messages when CICS does not have an open receive issued to the z/OS Communications Server. To determine whether this is happening, use CICS statistics (see "SNA statistics" on page 778) obtained following CICS shutdown. Compare the maximum number of RPLs that are posted in the shutdown statistics with the RAPOOL value in the SIT. If these values are equal, subpool 229 is probably being used to stage messages, and the RAPOOL value should be increased.

In some situations, the way in which the storage in the high private area is used might cause an S80A abend. There are at least two considerations:

- The use of MVS subpools 229 and 230 by access methods such as SNA.

  SNA and VSAM might 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).
- 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 can 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 to minimize fragmentation in the space above the region boundary. If CICS is a job step initiated in a previously started initiator's address space, the way in which LSQA and SWA storage is allocated might 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. The more effective use of the available storage might avoid S80A abends.

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 be changed easily. To create more space for the LSQA, SWA, and subpools 229 and 230, you might need to **decrease** the region size.

For more information about subpools and managing private storage allocation, see Virtual storage management in z/OS MVS Programming: Authorized Assembler Services Guide.

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 can 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 40 KB and 170 KB, depending on the complexity of the rest of the CICS address space.

The storage control blocks define the storage subpools in the private area, describing the free and allocated areas within those subpools. They can 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 in 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 9 KB for every DBCTL thread, up to the **MAXTHRED** value.

Scheduler work area (SWA):

The scheduler work area (SWA) 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 starts at 100 to 150 KB, and increases by about 1 to 1.5 KB per allocated data set.

A subset of SWA control blocks can, optionally, reside above 16 MB. 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 KB 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 the z/OS Communications Server 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 40 KB to 50 KB, allow 300 to 400 bytes per opened data set.

CICS DBCTL requires subpool 230 storage for DBCTL threads. Allow 3 KB for every DBCTL thread, up to the **MAXTHRED** value.

MVS storage above region

MVS storage above region is the storage that is left between the top of the region and the bottom of the high private area. Usually 200 KB to 300 KB of free storage is maintained to allow for use by the termination routines if these is an abend.

If this free storage is not enough for recovery termination management (RTM) processing, the address space might be terminated with a S40D abend that does not produce a dump.

This area can be very dynamic. As the high private area grows, it extends down into this area, and the CICS region can extend up into this area up to the value specified in **IEALIMIT**.

Splitting online systems: virtual storage

To increase the virtual storage available to a CICS system, you can split the system into two or more separate address spaces. Splitting a system can also provide higher availability, and you can use multiprocessor complexes to the best advantage because a system can then operate on each processor concurrently. Most CICS systems can be split.

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 run CICS in a single address space, you must then consider splitting the CICS workload into multiple address spaces. The new address spaces require more real storage, but the potential savings in virtual storage from splitting CICS regions are significant. You can split a CICS system by application function, by CICS function (such as a file owning or terminal owning region), or by a combination of the two functions.

Many installations find it convenient to split their CICS workloads into multiple independent address spaces, where the workload is easily definable and no resource sharing is required. If you can readily 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.

If you can split a CICS system completely, with no communication required between the two parts, you reduce overheads and planning. If the new systems must share data, programs, or terminals, you can use CICS intercommunication. You can use IPIC (IP interconnectivity) connections, ISC over SNA (intersystem communication over SNA) connections, or MRO (multiregion operation) to connect CICS regions to each other. For descriptions of the CICS intercommunication methods and the facilities that are available with each method, such as transaction routing and function shipping, see Intercommunication concepts and facilities in the *CICS Intercommunication Guide*.

You can also consider creating additional copies of a CICS region, and using CICS intercommunication to provide transaction routing between them. If additional virtual storage is needed, it is reasonable, for example, to split the AOR into two or more additional CICS copies. When you 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. Removing unused programs reduces the size of the relevant DSA.

CICS intercommunication uses additional processor cycles, and it can affect response time as well as processor time. The cost of intercommunication varies depending on the connection type (IPIC, MRO, or ISC over SNA), and on the intercommunication facilities that you use over that connection. For information about the performance considerations for different intercommunication methods and facilities, see Chapter 13, “CICS MRO, ISC, and IPIC: performance and tuning,” on page 171.

You might have to adjust certain parameters, such as **MXT**, when CICS systems are split. In an MRO system with function shipping, tasks of longer duration might also require further adjustment of **MXT** and other parameters (for example, file string numbers, virtual storage allocation).

If you plan to use MRO, consider sharing CICS code or application code using the MVS link pack area (LPA). Note that the LPA saves 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**, which tells CICS to search for the modules in the LPA. For further information about the use of the LPA, see “Using modules in the link pack area (LPA/ELPA).”

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.

There are a number of benefits of placing code in the LPA or ELPA:
- 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.

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, you might want 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 can 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.

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. All users with multiple CICS address spaces should put all eligible modules in the ELPA.

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 Installing CICS modules in the MVS link pack area.

Selecting aligned or unaligned maps

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. An aligned map is compiled with the AMAP option, and an unaligned one is compiled with the MAP option. A combination of aligned and unaligned maps can be used.

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 rincreases the size of the BMS DSECT, at worst by one padding byte per map data field, and marginally increases the internal path length 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 unaligned 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.

In CICS, BMS maps are always generated in groups (map sets). An entire map set must be defined as aligned or unaligned. Also, maps can be used by application programs written in various languages. In these cases, it is important to select the option that best suits the combination of programs and, if there is any requirement for both aligned and unaligned maps, select the ALIGNED option.

Avoid converting maps, for example, from aligned to unaligned, because changing the map DSECT also requires reassembly or recompilation of all application programs that reference it.

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. For more information, see the *Screen Definition Facility II Primer for CICS/BMS Programs*.

The importance of map alignment is demonstrated by inspecting programs that handle screens with many 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, use aligned maps where possible.

Defining programs as resident, nonresident, or transient

Programs, map sets, and partition sets can be defined as RESIDENT(NO | YES) and USAGE(NORMAL | TRANSIENT). Programs can be defined as RELOAD(NO | YES).

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 constrained, deleting first the programs that are used infrequently.

RESIDENT(YES) programs are not normally deleted. If NEWCOPY runs 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 deallocated until other (smaller) resident programs are loaded into it. This situation can occur if a resident program has increased its size since it was last loaded (before the last CICS shutdown). If the program in question is large, storage problems can occur because of the large amount of unused storage in the initial free area allocated for resident programs.

Because programs that are not in use are deleted on a least-recently-used (LRU) basis, define these programs 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.

Therefore, a much-used nonresident program is likely to remain resident anyway, while during periods of light usage, a resident program might be wasting the virtual storage it permanently occupies.

For programs written to run above the 16 MB line, specify EDSALIM large enough such that virtual storage is not a constraint.

If a program is 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.

You might define a program as RESIDENT for one of the following reasons:
- To avoid storage fragmentation, because all such programs are in a single block of storage (but not new copies of programs).
- For programs that deal with potential crises (for example, CEMT).

- Where there is heavy contention on the DFHRPL or dynamic program LIBRARYs. However, contention is usually handled by data set placement or other DASD tuning, or with use of MVS library lookaside to maintain program copies in an MVS dataspace.

Putting application programs above 16 MB

CICS keeps RMODE(ANY) application programs in the EDSA, which is in MVS extended virtual storage above 16 MB but below 2 GB (above the line). Work areas associated with the programs can also reside above the line.

It is possible to LINK or XCTL between 64-bit, 31-bit, and 24-bit addressing mode (AMODE) programs. You can convert programs to 31-bit or 64-bit addressing mode programs and move them above 16 MB but below 2 GB to the extended private area. Moving programs above 16 MB but below 2 GB frees that amount of virtual storage below 16 MB for other use.

See “Using modules in the link pack area (LPA/ELPA)” on page 138 for information about using programs from the LPA or extended link pack area (ELPA).

Using the ELPA is 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 that has 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, which has a 1 MB page size, rather than the UDSA, which has a 4 KB page size. This facility is useful where there is demand for virtual storage up to the 16 MB line and there is sufficient real storage. Because the reason for using virtual storage above the line is to make the space below 16 MB available for other purposes, there is an overall increase in the demand for real storage when programs are moved above 16 MB but below 2 GB.

When a COMMAREA is passed between programs running in different addressing modes, the following restrictions apply:

- A COMMAREA passed from an AMODE(31) program to an AMODE(24) program must be able to be processed by the AMODE(24) program, therefore it must not contain 31-bit addresses.
- A COMMAREA passed from an AMODE(64) program to an AMODE(31) program must be able to be processed by the AMODE(31) program, therefore it must not contain 64-bit addresses.
- A COMMAREA passed from an AMODE(64) program to an AMODE(24) program must be able to be processed by the AMODE(24) program, therefore it must not contain 64-bit or 31-bit addresses.

Programs that are to reside above the 16 MB line must be link-edited with the AMODE(31),RMODE(ANY) options on the MODE statement of the link-edit.

Allocation of real storage when using transaction isolation

When transaction isolation is active, there is a cost in terms of real storage. If insufficient real storage is allocated, paging problems can result, which then affect performance. The cost depends on the number of subspaces in use in the system, and the size of the **EDSALIM** parameter.

Because the page size of the EUDSA is 1 MB, the value of **EDSALIM** is likely to be very large for a CICS system that has transaction isolation active. This virtual storage needs to be mapped with page and segment tables using real storage, so an increase in the real storage usage can occur. In addition to the real storage used to map the virtual storage for the **EDSALIM** value, subspaces also require real storage. For example:

- Each subspace requires 2.5 pages, where a page means a 4 KB page of real storage.
- Assuming that each transaction in the system requires a unique subspace, (transaction definition TASKDATAKEY(USER) and ISOLATE(YES)), real storage required is the **MXT** value x 2.5 pages.
- If each transaction in the system requires a page of storage in the EUDSA (1 MB page), a page table is required to map the storage. Real storage is the **MXT** value x 1 page.
- A further three pages are required, so the total of real storage is the **MXT** value x (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 system that does not have transaction isolation active. The CICS requirement for real storage varies depending on the transaction load at any one time. As a guideline, each task in the system requires 9 KB of real storage. Multiply this number by the number of concurrent tasks that can be in the system at any one time (governed by the **MXT** system initialization parameter).

Limiting the expansion of subpool 229 using SNA pacing

Subpool 229 can 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. The PACING parameter controls the flow of traffic from the network control program (NCP) to the terminal and does not affect the processor activity as such. The VPACING parameter 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 z/OS Communications Server application program by another SNA logical unit without requiring that an acknowledgment (a pacing response) is returned. The host sends data path information units (PIUs) according to the definition of the VPACING parameter. 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 so 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, might cause a significant increase in host activity.

Normally, the VPACING parameter is implemented when a shortage of NCP buffers requires controlling the volume of flow between the host and the NCP. You can lessen the effect on the processor by increasing the VPACING parameter to a value that the NCP can 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 the PACING and VPACING parameters causes both response time degradation and increased processor activity, and increased network traffic.

Specify the PACING and VPACING parameters for all terminals to prevent a “runaway” transaction from flooding the SNA network with messages and requiring large amounts of buffer storage. If a transaction loops while issuing SEND commands to a terminal, IOBUF (CSA storage) and NCP buffers can fill up causing slowdowns and CSA shortage conditions.

Specify the PACING and VPACING parameters high enough so that normal data traffic can flow without being regulated, but excessive amounts of data are prevented from entering the network and impairing the normal flow of data.

For secondary to primary pacing, you must code in the following way:
- 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. Ensure that the device supports this form of pacing as specified in the component description manual for that device.

Chapter 8. CICS storage protection facilities: Performance and tuning

The facilities that are related to storage protection are storage protection, transaction isolation, and command protection. These facilities protect storage from user application code.

Storage protection
: Protects CICS code and control blocks from being overwritten accidentally by user applications.

Transaction isolation
: Offers protection against transaction data being overwritten accidentally 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.

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; that is, a pure terminal-owning region (TOR) or a pure file-owning region (FOR) (where no distributed program link (DPL) requests are function-shipped).

Transaction isolation and applications

When using transaction isolation, it is necessary to *activate* pages of storage to the allocated subspace of the task. Before the storage is activated to the subspace, it is fetch protected so that the task cannot access the storage. After the storage is activated to the subspace allocated to the task, the task has read and write access to the storage. CICS must activate user storage to a subspace every time that the user task invokes a GETMAIN command to get 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 16 MB line is activated in multiples of 4 KB. Storage above the line is activated in multiples of 1 MB. So a user task that runs completely above the line is more likely to require only one activate operation.

Link edit your programs by using RMODE(ANY) and define them as DATALOCATION(ANY). All transactions should be defined as TASKDATALOC(ANY), thus reducing the number of storage activations.

When you need to obtain storage below the line, you can improve performance by obtaining all the storage in one GETMAIN request, rather than several smaller GETMAIN requests. This also minimizes the number of storage activate operations.

For more information, see MVS subspaces in Developing applications.

Chapter 9. Tuning with Language Environment

When you run with Language Environment on CICS, there are several tuning actions you can take to optimize performance. If Language Environment is active in a CICS address space, the runtime libraries of the native language, such as COBOL or PL/I, are not needed. This means that CICS has a single interface to all the language run times.

For more information about Language Environment, see Programming languages and Language Environment in Developing applications.

Minimizing GETMAIN and FREEMAIN activity

One way to improve performance when you run programs with Language Environment is to reduce the number of GETMAIN and FREEMAIN requests required to manage the storage that Language Environment uses.

You can use the following system initialization parameters to minimize the number of GETMAIN and FREEMAIN requests that CICS performs on behalf of Language Environment:

- AUTODST
- RUWAPOOL

You can use these two options together in any combination.

To check the benefit of using these functions, run a CICS storage report to show the number of GETMAIN and FREEMAIN request in a region when either or both of the functions are active, and compare the results with previous runs.

AUTODST: Language Environment automatic storage tuning

You can optionally activate Language Environment's automatic storage tuning feature for CICS by setting the CICS system initialization parameter AUTODST to YES. When this function is active, Language Environment monitors each main program execution, and notes if any additional storage had to be allocated for the program while it was active.

At the end of each program execution, if any additional storage had to be allocated, Language Environment retains this information. Next time the program is executed, Language Environment increases the initial storage allocation to include this extra storage. This process helps to minimize the number of GETMAIN and FREEMAIN requests that CICS has to perform.

Automatic storage tuning is particularly helpful for programs that issue many dynamic calls, as such programs can easily exceed their initial storage allocations. It also removes the need to tune storage manually for individual COBOL programs.

However, you should be aware that once Language Environment has increased the initial storage allocation for a program, it is never decreased. If a program execution requires an unusually large amount of storage, perhaps because the user has activated a seldom-used function of the program, this amount of storage is allocated for all subsequent executions of the program. So in rare cases, you can find that automatic storage tuning leads to an excessive allocation of storage for some programs.

You can alter the behavior of the automatic storage tuning mechanism using the Language Environment storage tuning user exit CEECSTX. The user exit can enable or disable automatic storage tuning for a particular program, and you might find this useful if you have an application whose storage needs vary greatly between different executions. It can also provide the starting values for initial storage allocation, and you can use it to limit the maximum amount of storage that Language Environment will allocate during the automatic storage tuning process.

If the CEECSTX user exit was previously used as your Language Environment storage tuning method, you might find that the automatic storage tuning mechanism provides the same function, without the user exit. You need to decide which mechanism to use as your main storage tuning method, because when you are running CICS with automatic storage tuning, the CEECSTX user exit has limited function. Automatic storage tuning operates by monitoring storage allocations, whereas the storage tuning user exit CEECSTX monitors the actual storage used by the user application program. Despite this, automatic storage tuning incurs less overhead than the tuning method based on the CEECSTX exit. Also, automatic storage tuning provides tuning for each initial program invoked by a transaction, while the CEECSTX exit provides tuning for only those programs contained in the table that the exit uses as its input. This means that automatic storage tuning can provide a greater benefit by tuning the storage used by more programs.

For more information about CEECSTX, see z/OS Language Environment Customization.

RUWAPOOL: Run-unit work area pools

The system pathlength increases when a CICS application invoked by Language Environment issues an **EXEC CICS LINK** request. Repeated **EXEC CICS LINK** calls to the same program invoked by Language Environment result in multiple GETMAIN and FREEMAIN requests for run-unit work areas (RUWAs).

Using the system initialization parameter `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 programs invoked by Language Environment. This reduces the number of GETMAIN and FREEMAIN requests in tasks that perform many **EXEC CICS LINKS** to programs invoked by Language Environment.

For more information about the **RUWAPOOL** system initialization parameter, see RUWAPOOL.

Language Environment run time options for AMODE (24) programs

The default Language Environment runtime options for CICS are ALL31(ON) and STACK(ANY). This means all programs that require Language Environment must be capable of addressing 31-bit storage, that is, must be AMODE(31), when Language Environment is enabled.

For AMODE(24) programs to run in a Language Environment-enabled CICS region, you can specify ALL31(OFF) and STACK(BELOW) for those programs that must run below the 16 MB line. However, if you change these options globally so that all programs use them, large amounts of storage will be allocated below 16 MB, which might cause a short-on-storage condition. When the ALL31(OFF) option is used, Language Environment acquires some control blocks, for example the RUWA, both above and below the 16 MB line, and so additional GETMAIN and FREEMAIN requests are needed to manage the duplicate control blocks.

You do not need to specify ALL31(OFF) as long as the program in question is the **initial** program invoked by a transaction, because Language Environment acquires storage for the enclave (program) in the correct addressing mode automatically. The exception is an AMODE(31) program that dynamically calls an AMODE(24) program. In that situation, the dynamically called AMODE(24) program needs to specify ALL31(OFF).

Using DLLs in C++

When each dynamic link library (DLL) is first loaded, the cost of initialization can be determined by the size of writable static area required by the DLL. Initialization costs can be reduced by removing unnecessary items from the writable static area.

When using DLLs, you should consider the following:

- Specifying the #pragma variable (x,NORENT). This places some read-only variables such as tables in the code area.
- Specifying #pragma strings(readonly). This works for C code whose default is that literal strings are modifiable. C++ already has literal strings as read only by default.
- Examine the prelinker map to determine the large areas. If you find, for example, @STATICC, you have unnamed writable static objects such as strings or static variables.

Minimizing the time Language Environment spends writing dump output to transient data queue CESE

The Language Environment runtime option TERMTHDACT controls the type and amount of diagnostic output produced by Language Environment for an unhandled error.

Using TERMTHDACT(DUMP), TERMTHDACT(TRACE), TERMTHDACT(UADUMP), or TERMTHDACT(UATRACE) can create a significant overhead in a production environment. These settings can cause large amounts of traceback and Language Environment dump data to be written to the CESE transient data queue.

If a traceback or CEEDUMP is not needed by the application environment, use TERMTHDACT(MSG) to eliminate the performance overhead of writing formatted CEEDUMPs to the CICS transient data queue CESE. If the traceback or CEEDUMP is required by the application, specify the CICSDDS option of TERMTHDACT to direct the Language Environment diagnostic output to the CICS dump data set, rather than to the CESE transient data queue.

Chapter 10. Java applications: performance and tuning

You can improve the performance of your Java applications and the JVMs in which they run by analyzing and tuning your CICS regions.

For more information about improving the performance of your Java applications, see Improving Java performance in Improving performance.

For more information about using CICS statistics to manage and tune the Java workloads running in your CICS regions, see "JVM server statistics" on page 576.

Chapter 11. MVS and DASD: performance and tuning

Tuning CICS for virtual storage under MVS depends on several elements: z/OS systems tuning, z/OS Communications Server SNA tuning, CICS tuning, and VSAM tuning. Because tuning is a top-down activity, ensure that you have already made a vigorous effort to tune z/OS before tuning CICS.

Your main effort to reduce virtual storage constraint and to get relief is concentrated on reducing the life of the various individual transactions; in other words, shortening task life. Upgrading your z Systems hardware can be a fast path to shortening task life:

- 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.
- 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.
- 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.

Look for the following indicators in your z/OS system to see if you have a problem with I/O specifically:

- Service level objectives are missed.
- Users complain about response times.
- I/O indicators show signs of stress, or you see high DEV DLY or USG for an important workload directly in Monitor III reports.

For more information, see the section about analyzing I/O activity in the *z/OS Resource Measurement Facility Performance Management Guide*.

MVS provides storage isolation for an MVS 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 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 processor. (For an overall description of virtual storage, see "CICS virtual storage" on page 85.) Review the virtual storage effects of combining the following kinds of workload on a single-image MVS system:

1. CICS and a large number (100 or more) of TSO users

2. CICS and a large IMS system
3. CICS and 5000 - 7500 SNA LU.

By its nature, CICS requires a large private region that might not be available when the common requirements of the large system on these other subsystems are satisfied. If, after tuning the operating system, SNA, 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 SNA 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 typically 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 the DFHRPL concatenation or dynamic LIBRARY concatenation.

The rest of this section describes techniques that you can use to improve the performance of CICS under MVS.

Chapter 12. Networking and the z/OS Communications Server: performance and tuning

The performance of your SNA network and logical units (LUs) can be tuned in a number of different ways.

This section includes the following topics:

- https://ut-ilnx-r4.hursley.ibm.com/ts42_latest/help/topic/com.ibm.cics.ts.performance.doc/topics/dfht34d.html
- "Setting the size of the receive-any input areas" on page 157
- "Setting the size of the receive-any pool" on page 158
- "Using the MVS high performance option with SNA" on page 160
- "Adjusting the number of transmissions in SNA transaction flows" on page 161
- Using SNA chaining to segment large messages
- "Limiting the number of concurrent logon and logoff requests" on page 163
- "Adjusting the terminal scan delay" on page 164
- "Compressing output terminal data streams" on page 167
- Turning automatic installation of terminals

Setting the size of the terminal input and output area

The **IOAREALEN** attribute of a TYPETERM RDO resource definition specifies the size of the terminal input and output area that is to be passed to a transaction. The size of the TIOA can also be specified by the **TIOAL** parameter in the DFHTCT TYPE=REMOTE macro, if macro resource definition has been used.

The syntax for the **IOAREALEN** attribute in a TYPETERM RDO resource definition is *({0|value1},{0|value2})*. This setting is used only for the first input message for all transactions. 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.

If you specify *ATI(YES)*, you must specify an **IOAREALEN** value of at least one byte.

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 uses *value2*. If the input message size exceeds *value2*, the node abnormal condition program sends an exception response to the terminal.

Limitations

Real storage can be wasted if the **IOAREALEN** (*value1*) value, or the value for the **TIOAL** parameter in the DFHTCT TYPE=REMOTE macro, is too large for most terminal inputs in the network. However, 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.

Suggestions

Set **IOAREALEN** (*value1*) or **TIOAL** to a value that is slightly larger than the average input message length for the terminal. The maximum value that can be specified for **IOAREALEN** or **TIOAL** is 32767 bytes.

If a value of nonzero is required, specify 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 the z/OS Communications Server, you can specify two values if inbound chaining is used. The first value is the length of the normal chain size for the terminal and the second value is 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 following example:

```
Where x is any number of bytes, the following applies.

Without chain assembly:

If the TIOA size is specified as 20x
and the message length is       15x
then the TIOA acquired is       20x

If the TIOA size is specified as 20x
and the message length is       25x
then the TIOA acquired is       25x

With chain assembly:

If Value1 size is               20x
and Value2 size is              25x, then
if the length of a message is   15x
the TIOA acquired is            20x
and if the message length is    22x
the TIOA acquired is            25x
```

Figure 18. Message length and terminal input and output area length

Avoid specifying a *value1* that is too large, 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.

Avoid specifying a *value1* that is too small, because 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 make it 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.

Monitoring

RMF and NetView Performance Monitor (NPM) can be used to show storage usage and message size characteristics in the network.

Setting the size of the receive-any input areas

The system initialization parameter, **RAMAX**, specifies the size in bytes of the I/O area that is to be allocated for each SNA receive-any operation. You can use the **RAMAX** system initialization parameter in any networks that use the z/OS Communications Server SNA access method for LUs.

These storage areas are called receive-any input areas (RAIAs) and are used to receive the first terminal input for a transaction from the SNA. All input from SNA comes in request/response units (RUs).

Storage for the RAIAs, which is above the 16 MB line, is allocated by the CICS terminal control program during CICS initialization. This storage 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

SNA attempts to put any incoming RU into the initial receive-any input area, which has the size of **RAMAX**. If this area is not large enough, SNA creates a message indicating the problem and stating 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. It is a limit against which CICS compares the indication from SNA of the overall size of the RU. If there is more, it is saved by SNA, and CICS gets the rest in a second request.

With a small **RAMAX**, you reduce the virtual storage taken up in RAIAs. However, you risk more processor usage in SNA tries again 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 value, 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 these devices. It makes sense for **RAMAX** to be at least as large as the largest **SENDSIZE** attribute for frequently used terminals.

Limitations

Real storage can be wasted with a high **RAMAX** value. If the **RAMAX** value is set too low, extra processor time is needed to acquire additional buffers to receive the remaining data.

Suggestions

Set **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. Because most inputs are 256 bytes, this size is the default value specified.

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 most 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** to be set based on their overall chain length, but only on the size of the constituent RUs.

Do not specify a **RAMAX** value that is less than the RUSIZE (from the CINIT) for a pipeline terminal because pipelines cannot handle over-length data.

Receive-any input areas are taken from a fixed-length subpool of storage. A size of 2048 might appear to be adequate for two such areas to fit on one 4 KB page, but only 4048 bytes are available in each page, so only one area fits on one page. Defining a size of 2024 ensures that two areas, including page headers, fit on one page.

Monitoring

The size of RUs or chains in a network can be identified with an SNA line or buffer trace.

Setting the size of the receive-any pool

The **RAPOOL** system initialization parameter specifies the number of concurrent receive-any requests that CICS is to process from the z/OS Communications Server for SNA.

RAPOOL determines how many receive-any buffers there are at any time. Therefore, if the z/OS Communications Server for SNA has a lot of input simultaneously, it enables the z/OS Communications Server to put all the messages directly into CICS buffers rather than possibly having to store them 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 RAPOOL in the *CICS 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 SNA access method and is passed to CICS if CICS has a receive-any request outstanding.

For each receive-any request, an SNA request parameter list (RPL), a receive-any control element (RACE), and a receive-any input area (RAIA) are set aside. The RAIA value is specified by **RAMAX** (see "Setting the size of the receive-any input areas" on page 157 for RAIA considerations). The total area set aside for SNA receive-any operations is:

(maximum RAIA size + RACE size + RPL size) * RAPOOL

If *HPO=YES*, both RACE and RPL are above the 16 MB line.

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 are specified in the **RAPOOL** value can be processed in one dispatch of terminal control. This situation happens when a receive-any request completes before the terminal control program has finished processing and there are additional messages from SNA.

The specified pool is used only for SNA receive-any processing of the first terminal message in a transaction or the first input to start a task. **RAPOOL** does not affect further inputs for conversational tasks or output. Additional inputs are processed with SNA receive-specific requests.

SNA 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.

The significance of **RAPOOL** depends on the environment of the CICS system. For example, if HPO is used then **RAPOOL** is significant.

Limitations

If the **RAPOOL** value is set too low, terminal messages might not be processed in the earliest dispatch of the terminal control program, causing transaction delays during high-activity periods. For example, if you use the default value and five terminal entries need to startup tasks, three tasks might be delayed for at least the time required to complete the SNA receive-any request and copy the data and RPL. In general, set no more than 5 to 10% of all receive-any processing at the **RAPOOL** ceiling, with none at the **RAPOOL** ceiling if there is sufficient storage.

If the **RAPOOL** value is set too high, excessive virtual storage might be used, but does not affect real storage because the storage is not page-fixed and is therefore paged out.

Suggestions

In some cases, it might be more economical for SNA to store the occasional peak of messages in its own areas rather than for CICS to have many RAIAs, which are unused most of the time.

Furthermore, there are situations where CICS reissues a receive-any request as soon as it finds one satisfied. It uses the same element over and over again in order to bring in any extra messages that are in SNA.

CICS maintains a z/OS Communications Server **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 System Definition Guide* for more information about these system initialization parameters.

Code **RAPOOL** with the number of fixed request parameter lists (RPLs) that you require. When it is not at the **MXT** value, 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 "Adjusting the terminal scan delay" on page 164) in the system initialization table (SIT). Initially, for non-HPO systems, set **RAPOOL** to 1.5 times your peak *local* transaction rate per second plus the autoinstall rate. This value can then be adjusted by analyzing the CICS SNA statistics and by resetting the value to the maximum RPLs reached. The **RAPOOL** value does not include MRO sessions, so set this value to a low number in application-owning or file-owning regions (AORs or FORs).

For HPO systems, a small value (<= 5) is typically sufficient if specified through *value2* in the **RAPOOL** system initialization parameter. For example, `RAPOOL=20` is specified as either `RAPOOL=(20)` or `RAPOOL=(20,5)` to achieve the same effect.

Monitoring

The CICS SNA 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 can be greater than the **RAPOOL** value if the terminal control program is able to reuse an RPL during one dispatch. See "Interpreting z/OS Communications Server statistics" on page 778 for more information.

Using the MVS high performance option with SNA

The MVS high performance option (HPO) can be used for processing SNA requests. The purpose of HPO is to reduce the transaction path length through the z/OS Communications Server.

The use of HPO and supervisor calls (SVCs) are specified in the system initialization table (SIT). If the default SVC numbers are acceptable, no tailoring of the system is required.

Effects

HPO bypasses some of the validating functions performed by MVS on I/O operations, and implements service request block (SRB) scheduling. This bypass shortens the instruction path length and allows some concurrent processing on MVS images for the z/OS Communications Server operations because of the SRB scheduling. This effect makes HPO useful in a multiprocessor environment, but not in a single processor environment.

Limitations

HPO requires CICS to be authorized. 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. An expense of HPO might be the potential security exposure that arises because of a deficiency in validation.

Suggestions

All production systems with vetted applications can use HPO. It is application-transparent and introduces no function restrictions while providing a reduced pathlength through the z/OS Communications Server. For z/OS Communications Server, the reduced validation does not induce any integrity loss for the messages.

Monitoring

There is no direct measurement of HPO. One method to check whether 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 might not see much difference. Another way to check whether it is working is that you might see a small increase in the SRB scheduling time with HPO turned on.

RMF can give general information about processor usage. An SVC trace can show how HPO was used.

Take care when 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 the z/OS Communications Server. Untested code might possibly corrupt the control blocks that CICS passes to the z/OS Communications Server, and unvalidated applications can lead to security exposure.

Adjusting the number of transmissions in SNA transaction flows

Within CICS, the **MSGINTEG** and **ONEWTE** options can be used to control the communication requests and responses that are exchanged between the terminals in a network and the z/OS Communications Server and NCP communication programs. These options can be used in all CICS systems that use the Communications Server

With resource definition online (RDO), protection can be specified in the PROFILE definition with the **MSGINTEG**, and **ONEWTE** options. The **MSGINTEG** option is used with SNA logical units (LU) only. See PROFILE resource definitions in the *CICS IMS Database Control Guide* for more information about defining a PROFILE resource.

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 message receipt 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 message integrity (**MSGINTEG**) causes the sessions for which it is specified to operate in definite response mode.

In normal cases, the session between CICS and a terminal operates in exception response mode.

You therefore have the following options:

- Not specifying **MSGINTEG**
- Specifying **MSGINTEG** (which asks for definite response to be forced)

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 one write operation (**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.

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.

When **MSGINTEG** is specified, the TIOA remains in storage until the response is received from the terminal. This option might increase the virtual storage requirements for the CICS region because of the longer duration of the storage needs.

Monitoring

You can monitor the use of the **MSGINTEG** and **ONEWTE** options from a Communications Server trace by examining the exchanges between terminals and CICS and, in particular, by examining the contents of the request/response header (RH).

Using SNA chaining to segment large messages

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. Chaining can be used in systems that use z/OS Communications Server SNA LUs of types that tolerate chaining.

Chaining characteristics are specified with the **SENDSIZE**, **BUILDCHAIN**, and **RECEIVESIZE** attributes.

The hardware requirements of each terminal normally dictate the input chain size and characteristics. The **BUILDCHAIN** and **RECEIVESIZE** attributes have default values that depend on device attributes. The size of an output chain is specified by the **SENDSIZE** attribute.

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 processing effects of output chaining. A value of 0 or 1536 is required for local devices of this type.

If you specify the **SENDSIZE** attribute for intersystem communication (ISC) sessions, this attribute must match the **RECEIVESIZE** attribute in the other system. The **SENDSIZE** attribute or **TCT BUFFER** operand controls the size of the SNA element that is to be sent, and the **RECEIVESIZE** must 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 large inbound elements that exceed a maximum of 32 KB, you cannot use the **BUILDCHAIN** attribute or **CHNASSY** operand. You must use multiple individual RUs, which extends the transaction life in the system.

Limitations

If you specify a low **SENDSIZE** value, this setting causes additional processing. Real and virtual storage are used to break the single logical message into multiple parts.

Chaining might be required for some terminal devices. Output chaining can cause flickering on display screens, which users might find disruptive. Chaining also causes additional I/O processing effects between the z/OS Communications Server and the NCP by requiring additional z/OS Communications Server subtasks and STARTIO operations. These effects are eliminated with applicable ACF/SNA releases by using the large message performance enhancement option (LMPEO).

Suggestions

The **RECEIVESIZE** value for IBM 3274-connected display terminals is 1024 and for IBM 3276-connected display terminals it is 2048. These values give good line characteristics while keeping processor usage to a minimum.

Monitoring

Use of chaining and chain size can be determined by examining a z/OS Communications Server trace. You can also use the CICS internal and auxiliary trace facilities, where the VIO ZCP trace shows the chain elements. Some network monitoring tools such as NetView Performance Monitor (NPM) give this data.

Limiting the number of concurrent logon and logoff requests

The **OPNDLIM** system initialization parameter defines the number of concurrent z/OS Communications Server logon and logoff requests that are to be processed by CICS. This parameter can be used in CICS systems that use the z/OS Communications Server as the terminal access method.

The **OPNDLIM** 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.

This 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. For more information about this parameter, see OPNDLIM system initialization parameter in Reference -> System definition.

Each concurrent logon and logoff requires storage in the CICS dynamic storage areas for the duration of that processing.

Effects

When logons occur automatically with either the CICS CONNECT=AUTO facility or the z/OS Communications Server LOGAPPL facility, large numbers of logons can occur at CICS startup or restart times.

The LOGAPPL facility offers two advantages if an automatic logon facility is required: it requires approximately 3500 bytes less storage in the z/OS Communications Server than the CONNECT=AUTO facility, and it logs terminals back on to CICS each time the device is activated to the z/OS Communications Server, rather than only at CICS initialization.

Limitations

If the value specified for **OPNDLIM** is too low, real and virtual storage requirements are reduced within CICS, and the z/OS Communications Server buffer requirements might be cut back, but session initialization and terminations take longer.

Suggestions

Use the default value initially and adjust if statistics indicate that too much storage is required in your environment or that the startup time is excessive.

Set **OPNDLIM** to a value not less than the number of logical units (LU) connected to any single z/OS Communications Server line.

Monitoring

Logon and logoff activities are not reported directly by CICS or any measurement tools, but can be analyzed using the information given in a z/OS Communications Server trace or z/OS Communications Server display command.

Adjusting the terminal scan delay

The terminal scan delay (**ICVTSD**) system initialization parameter determines the frequency with which CICS attempts to process terminal output requests.

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**.

In reasonably active systems, a nonzero **ICVTSD** virtually replaces ICV, because the time to the next terminal control table (TCT) full scan (non-SNA) or sending of output requests (SNA) is the principal influence on wait duration of the operating system.

The **ICVTSD** parameter can be used in all except very low-activity CICS systems.

In general, the **ICVTSD** value defines the time that the terminal control program must wait to process the following requests:
- Non-SNA LU I/O requests with WAIT specified
- Non-SNA output deferred until task termination
- Automatic transaction initiation (ATI) requests
- SNA LU 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-SNA 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 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.

Effects in SNA networks

In SNA networks, a low **ICVTSD** value does not cause full TCT scans, because the input from or output to SNA LU is processed from the activate queue chain, and only those terminal entries are scanned.

Request batching reduces processor time at the expense of longer response times. On CICS SNA systems, it influences how quickly the terminal control program completes SNA request processing, especially when the MVS high performance option (HPO) is being used.

With SNA LUs, 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. Thus, there might 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. 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 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 SNA for an extended period, the life of a transaction can be extended. Storage is kept allocated for that task for longer periods, potentially increasing the amount of virtual storage required for the total CICS dynamic storage areas. Setting **ICVTSD** to zero can overcome this effect

Effects in non-SNA networks

ICVTSD is the major control on the frequency of full TCT scanning of non-SNA LUs. In active systems, a full scan is done approximately once every **ICVTSD** period. The average extra delay before sending an output message is about half this period.

In non-SNA networks, partial scans occur for other reasons, such as an input arriving from a LU, 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-SNA networks.

CICS scans application tasks first, unless there is a scan driven by **ICVTSD**. In a highly used system, input and output messages might be unreasonably delayed if too large a **ICVTSD** value is specified.

Effects in all networks

The **ICVTSD** parameter can be changed in the system initialization table (SIT) or through JCL parameter overrides. If you have virtual storage constraint problems, reduce the value specified in **ICVTSD**. A value of zero causes the terminal control task to be dispatched most frequently. If you also have many non-SNA LUs, this value might increase the amount of nonproductive processor cycles. A value of 100—300 ms might be more appropriate for that situation. In a pure SNA environment, however, the processing effect is not significant, unless the average transaction has a short pathlength. Set **ICVTSD** to zero for a better response time and best virtual storage usage.

Limitations

In z/OS Communications Server (for SNA) systems, a low value adds the processing effect 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, which can be a significant consideration.

A low, nonzero value of **ICVTSD** can cause CICS to be dispatched more frequently, which increases the processing effect of performance monitoring.

Suggestions

Set **ICVTSD** to a value less than the region exit time interval (ICV), which is also in the system initialization table. Use the value of zero in an environment that contains only SNA LUs and consoles, unless your workload consists of many short transactions.

Entering `ICVTSD=0` in an SNA LU-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 effect by causing CSTP to be dispatched regularly. For non-SNA systems, specify the value of zero only for small networks (1 - 30 terminals).

For almost all systems that are not “pure” SNA, set the range somewhere in the region of 100 ms to 1000 ms. **ICVTSD** can be varied from 300 - 1000 ms without a significant effect on the response time, but increasing the value decreases the processor activity effect. An **ICVTSD** larger than 1000 ms might 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 set the value below 250 ms, any improvement in response time is likely to seem negligible to the user and would have an increased effect on processor usage.

The absolute minimum level, for systems that are not “pure” SNA, is approximately 250 ms, Or, in high-performance, high-power systems that are “pure” SNA, the level is 100 ms.

Monitoring

Use RMF to monitor task duration and processor requirements. The dispatcher domain statistics reports the value of **ICVTSD**.

Compressing 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.

For z/OS Communications Server for SNA devices, the global user exit used to compress terminal messages is **XZCOUT1**. For programming information, see SNA working-set module exits (XZCIN, XZCOUT, XZCOUT1, and XZIQUE).

This compression technique can produce a dramatic improvement in response times if the proportion of characters not needed is large, because telecommunication links are typically 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. Using a compression exit reduces the storage requirements of SNA and reduces line transmission time.

Suggestions

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 quickly on some types of 3270 cluster controller. In some cases, alternatives can give superior performance. For example, instead of sending a repeat-to-address sequence for a series of blanks, consider sending an ERASE and then set-buffer-address sequences to skip over the blank areas. This method 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 method eliminates the need for those characters to be transmitted back to the processor on the next input message, but review application dependencies on those fields before you try this approach.

There might be other opportunities for data compression in individual systems, but you need to investigate the design of those systems thoroughly before you can implement them.

Monitoring

The contents of output terminal data streams can be examined in an SNA trace.

Tuning 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 determined by the length of the CINIT request unit, which varies for different LU types. For a typical autoinstall request from an LU 6.2 terminal, the amount of dynamic virtual storage obtained is 120 -250 bytes.

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

The **AIQMAX** 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

The **AIRDELAY** 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 delay 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 reinstall the autoinstalled terminal entries from the global catalog during emergency restart. In this case, CICS does not write the terminal entries to the catalog while the terminal is being autoinstalled. This setting 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 might 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 global catalog data set during emergency restart. Thus, if you have many autoinstalled terminals, the restart time can be improved when restart delay is set to zero.

Normal shutdown
: CICS deletes AI terminal entries from the global catalog data set during normal shutdown unless they were not cataloged (*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 setting can reduce normal shutdown time.

You must consider 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 typically small.

The delete delay parameter

The **AILDELAY** 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 setting 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 the lifetime 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 reused when the terminal logs back on. This setting can eliminate the activities of:
- Deleting the terminal entry in virtual storage
- An I/O to the catalog and recovery log
- Rebuilding 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 reused when the terminal logs back on. This delay can save the processing effect 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 exist concurrently (see “Using transaction classes (MAXACTIVE) to control transactions” on page 73).
2. By using the CATA and CATD transactions to install and delete autoinstall terminals dynamically. If you have many 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, 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 setting 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 z/OS Communications Server to stop passing LOGON and BIND requests to CICS. z/OS Communications Server holds such requests until CICS indicates that it can accept further LOGONs and BINDs (occurs when CICS has processed a queued autoinstall request).

Suggestions

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 the value reached during normal operations.

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 affected by the definition of the global catalog (DFHGCD). The default buffer specifications used by VSAM might not be sufficient in a high activity system.

Because a considerable number of messages are sent to transient data during logon and logoff, consider the performance of these output destinations.

Monitoring

Monitor the autoinstall rate during normal operations by inspecting the autoinstall statistics regularly.

Chapter 13. CICS MRO, ISC, and IPIC: performance and tuning

Multiregion operation (MRO), intersystem communication over SNA (ISC over SNA), and IP interconnectivity (IPIC) connections enable CICS systems to communicate and share resources with each other. Performance is influenced by the intercommunication facilities that you use with the connection and by your management of the connection.

These CICS intercommunication facilities are available using MRO, and ISC over SNA, and IPIC connections:

- Function shipping
- Distributed transaction processing
- Asynchronous processing
- Transaction routing
- Distributed program link

For descriptions of the CICS intercommunication methods and facilities, see Getting started with intercommunication in Getting started.

CICS ISC/IRC statistics show the frequency of use of intercommunication sessions and mirror transactions. The z/OS Communications Server SNA trace, an SVC trace, and RMF give additional information.

If each transaction makes a number of intercommunication requests, function shipping generally incurs the most processor usage. 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 facilities for 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 Concepts and design considerations in the *CICS Distributed Transaction Programming Guide*.

Transaction routing, in most cases, involves one input and one output between systems, and the additional processor usage is minimal.

MRO

Multiregion operation (MRO), in general, causes less processor usage than intersystem communication (ISC) because the SVC pathlength is shorter than that through the multisystem networking facilities of SNA. CICS MRO provides a long-running mirror transaction and fastpath transformer program to further reduce processor usage.

Ensure that you have a sufficient number of MRO sessions defined between the CICS systems to take your expected traffic load. The increased cost in real and virtual storage is minimal, and task life is reduced, so the probable overall effect is to save storage. Examine the ISC/IRC statistics (see “ISC/IRC system and mode entry statistics” on page 535) to ensure that no allocates have been queued; also ensure that all sessions are being used. However, the definition of too many MRO sessions can unduly increase the processor time used to test their associated ECBs.

If you want only transaction routing with MRO, the processor usage is relatively small. The figure is release- and system-dependent (for example, it depends on whether you are using cross-memory hardware), but you can assume a total cost somewhere in the range of 15 - 30 KB 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 typically each transaction has many more inter-CICS flows. The cost depends greatly on the disposition of resources across the separate CICS systems.

MRO can affect response time as well as processor time. Delays occur in getting requests from one CICS system to the next. These delays arise because CICS terminal control in either CICS system has to detect any request sent from the other, and then has to process it. In addition, if you have a uniprocessor, MVS has to arrange dispatching of two CICS systems and that must imply extra WAIT/DISPATCH processor usage and delays.

Specify the system initialization parameter **`MROLRM=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.

When you use MRO, you can eliminate some processor usage for SVC processing 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, which can also be a benefit. Note, however, that MVS requires that an address space using cross-memory services be nonswappable.

ISC

For situations where ISC is used across MVS images, consider using XCF/MRO. CICS uses the MVS cross-system coupling facility (XCF) to support MRO links between MVS images for transaction routing, function shipping, and distributed program link. You can also use XCF/MRO for distributed transaction processing, if the LU6.1 protocol is adequate for your purpose. XCF/MRO consumes less processor resources than ISC.

You can prioritize ISC mirror transactions. The CSMI transaction is for data set requests, CSM1 is for communication with IMS systems, CSM2 is for interval control, CSM3 is for transient data and temporary storage, and CSM5 is for IMS DB requests. If one of these functions is particularly important, you can prioritize it over 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, you can control them with the SNA VPACING facility. Specifying multiple sessions (SNA parallel sessions) increases throughput by allowing multiple paths between the systems. With CICS, you can specify an SNA class of service (COS) table with LU6.2 sessions, which can prioritize ISC traffic in a network.

Interregion communication performance costs with MRO and ISC

Using the tables in these topics, you can compare the relative processing times of particular CICS API calls, and examine some of the other factors that affect overall processing times. These tables can help you make decisions concerning application design when you are considering performance. To calculate a time for a transaction, find the entries appropriate to your installation and application, and add their values together.

Before you work with these numbers, be aware of the following considerations:

- The cost per call is documented in 1 K or millisecond instruction counts taken from a tracing tool used internally by IBM. Each execution of an instruction has a count of 1. No weighting factor is added for instructions that use more machine cycles than others.
- Because the measurement consists of tracing a single transaction within the CICS region, any wait, for example a wait for I/O, results in a full MVS WAIT. This cost has been included in the numbers reported in this document. On a busy system the possibility of taking a full MVS WAIT is reduced because the dispatcher has a higher chance of finding more work to do.
- When judging performance, the numbers in this information should not be compared with those published previously, because a different methodology has been used.

Transaction routing performance costs

MRO XM	MRO XCF (through CTC)	MRO XCF (through CF)	ISC LU6.2
37.0	43.0	66.0	110.0

Function shipping performance costs (MROLRM=YES)

Type	MRO XM	MRO XCF (through CTC)	MRO XCF (through CF)
Initiate/terminate environment	13.2	13.2	13.2
Each function shipping request	9.0	23.4	48.4
Sync point flow	9.0	23.4	48.4

Notes:

- These costs relate to CICS systems with long-running mirrors.
- ISC LU6.2 does not support `MROLRM=YES`.
- The cost of session allocation, initiation of the mirror transaction, stopping the mirror transaction, and session deallocation is included in the initiate/terminate environment.

For example, if you migrate from a local file access to MRO XM and request 6 function ships per transaction, the additional cost is calculated as follows:

13.2(Initiate/End) + (6(requests)*9.0(Request cost)) + 9.0(Sync point) = 76.2

Function shipping performance costs (MROLRM=NO)

Without long-running mirrors, each function ship read request incurs the cost of session allocation and mirror initialization and termination. However, the first change to a protected resource (for example, a READ UPDATE or a WRITE) causes the session and mirror to be held until a sync point.

MRO XM	MRO XCF (through CTC)	MRO XCF (through CF)	ISC LU6.2
21.4	35.0	59.9	115.0

IPIC

The CICS-supplied mirror program DFHMIRS is defined as a threadsafe program. For supported CICS facilities, over IPIC connections only, the remote CICS region uses a threadsafe mirror transaction and runs the request on an L8 open TCB whenever possible. For threadsafe applications that issue commands for functions on remote CICS systems using IPIC connections, the reduction in TCB switching improves application performance compared to other intercommunication methods. The use of open TCBs also provides significant potential throughput improvements between CICS regions.

For some applications, the performance benefits of using long-running mirrors can also be significant. IPIC supports the MIRRORLIFE attribute of the IPCONN, which can improve efficiency and provide performance benefits by specifying the lifetime of mirror tasks and the amount of time a session is held.

IPIC supports threadsafe processing for the LINK command between CICS TS 4.2 or later regions. If you are using a threadsafe program that makes DPL requests that are transmitted to another region using IPIC connections, you might benefit from improved performance by changing your dynamic routing program to be coded to threadsafe standards. IPIC supports the following DPL calls:

- Distributed program link (DPL) calls between CICS TS 3.2 or later regions.
- Distributed program link (DPL) calls between CICS TS and TXSeries® Version 7.1 or later.

Function shipping file control, transient data, and temporary storage requests over an IPIC connection provides CICS application programs with the ability to run without regard to the location of the requested resources. Function shipping of file control and temporary storage requests using IPIC connections is threadsafe between CICS TS 4.2 or later regions. Function shipping of transient data requests using IPIC connections is threadsafe between CICS TS 5.1 or later regions. Any global user exit programs that are called in the remote CICS region for file control, transient data, and temporary storage requests must be enabled as threadsafe programs for the best performance.

For file control requests that are function shipped using IPIC connectivity, to gain the performance benefits of the open transaction environment, you must specify the system initialization parameter **FCQRONLY=NO** in the file-owning region.

Managing queues for intersystems sessions

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 memory 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 while 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 exists, 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 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.

XZIQUE exit for managing MRO and APPC intersystem queues in the *CICS 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

You can use connection statistics to detect problems in a CICS intersystem environment.

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 enough sessions defined?” on page 538 has more advice on setting the correct 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.

"Interpreting ISC/IRC system and mode entry statistics" on page 535 also contains an explanation of these statistics, and other connection statistics.

Ways of approaching the problem and recommendations

The queue limit mechanism can 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 of the MXT slots in the system. You can also use the MAXACTIVE setting of a TRANCLASS definition if you can segregate your transactions into classes that correspond to the remote regions they require.

To ensure free availability during normal running, provide a sufficient number of intersystems sessions. 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 are used, the connections statistics show the number of times allocates were queued compared with the total number of requests.

Even in a system that has no problems, there are significant variations in the numbers of transactions that are active at any time, and the actual peak number might 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 queuing 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 large problem, the detection mechanism is insensitive. If there are always queues in the system, you might experience false diagnosis.

You should set the queue limit to a number that 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, design your own method—using ZXIQUE—of controlling queue lengths so that the allocation of queue slots to connections is more dynamic.

The MAXQTIME parameter can be set to reflect the maximum wait time expected of users for responses in case of potential problems. The MAXQTIME parameter should not be set at a low value in combination with a queue limit that is low, because this leads to a 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, 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 due to variations in response time of the remote system, the parameter may be too low, or the QUEUELIMIT value too low.

Using transaction classes DFHTCLSX and DFHTCLQ2 to control storage use

Use DFHTCLSX and DFHTCLQ2 in RDO group DFHISCT to control the amount of storage used by CICS to run the CLS1, CLS2, and CLQ2 transactions.

Effects

These tasks execute the activities needed to acquire an APPC conversation (CLS1/2), and to resynchronize units of work for MRO and APPC connections (CLQ2). Typically there are not many tasks, and they need no control. However, if your CICS system has many connection definitions, these connections might be acquired simultaneously as a result of initializing the system at startup, or as a result of a **SET VTAM OPEN** or **SET IRC OPEN** command.

Note: VTAM is now z/OS Communications Server.

How implemented

The system definitions are optional. Install resource group DFHISCT to activate them. As supplied, the **MAXACTIVE** parameter in the DFHTCLSX and DFHTCLQ2 is 25. This value gives sufficient control to prevent the system reaching a short-on-storage situation. Tasks CLS1 and CLS2 each require 12 KB of dynamic storage, and CLQ2 tasks require up to 17 KB. Do not set the purge threshold to a non-zero number and do not set the **MAXACTIVE** parameter to 0. Both values might prevent CICS from running tasks necessary to intersystems functions.

Do not set the **MAXACTIVE** value too low, because network delays or errors might cause one of the tasks in the TCLASS to wait and block the use of the TCLASS by succeeding transactions. Setting a low value can also extend shutdown time in a system with many connections.

Controlling the length of the terminal input/output area (SESSIONS IOAREALEN) for MRO sessions

For MRO function shipping, the SESSIONS definition attribute, **IOAREALEN**, is used. This attribute 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 16 MB line.

The **IOAREALEN** value controls the length of the TIOA that is used to build a message transmitted to the other CICS system (that is, an outgoing message). You can specify two values (value1 and value2). Value1 specifies the initial size of the TIOA to be used in each session that is 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 uses value2 whenever the size of the message exceeds value1.

A value of zero causes CICS to get a storage area exactly the size of the outgoing message, plus 600 bytes for CICS requirements. If the IOAREALEN value is not specified, it defaults to 4 KB.

Where useful

The **IOAREALEN** attribute can be used in the definition of sessions for either MRO transaction routing or function shipping. For MRO transaction routing, the value determines the initial size of the TIOA, whereas in the MRO function shipping environment, the value presents some tuning opportunities.

Limitations

If the **IOAREALEN** value is too large for most messages transmitted on your MRO link, real and virtual storage might be wasted. If IOAREALEN is smaller than most messages, or zero, excessive FREEMAIN and GETMAIN requests might occur, resulting in additional processor requirements.

Recommendations

For optimum storage and processor utilization, make **IOAREALEN** 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 600 bytes for CICS requirements and round up the total to a multiple of 64 bytes. A multiple of 64 bytes (or less) minus 600 bytes for CICS requirements ensures a good use of operating system pages.

How implemented

The TIOA size can be specified in the **IOAREALEN** attribute of the SESSIONS definition.

Batching requests (MROBTCH)

Certain events in a region can be accumulated in a batch before 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
- Request completion carried out as a subtask on the CO TCB (mostly VSAM, and 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 of changing the default value of MROBTCH

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. For example, for n=2, 50% savings might be achieved, for n=3, 66% savings, or for n=6, 83% savings.
- An average cost of (n+1)/2 times the average arrival time for each request batched.

- Increased response time might 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 queuing for a busy resource. Using a low MROBTCH value greater than one might 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.

Setting a suitable batch value

Depending on the amount of response time degradation you can afford, you can set MROBTCH to different values. Use the CICS-SM perspective of the CICS Explorer (**Operations** > **Regions view** > **Region attributes** > **MRO Batch requests**) or use **EXEC CICS SET SYSTEM MROBATCH** to arrive at a suitable batch value for a given workload.

For programming information about the **EXEC CICS** system programming commands, see System commands in Reference > System programming.

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)

The 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, 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 the front-end region prevents the mirror task in the back-end region from being terminated after syncpoint. The mirror task in the back-end region will only be terminated when the front-end task terminates.

Use of **MROFSE=YES** in the front-end region is not recommended when long-running tasks may be used to function-ship requests. This is because a SEND session will be unavailable for allocation to other tasks when unused. It might also prevent the connection from being released when contact has been lost with the back-end region, until the task terminates or issues a function-ship request.

Controlling the 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 can 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 might require that redundant shipped terminal definitions are not permitted to persist in an AOR.

The CICS-supplied transaction CRMF periodically scans the shipped terminal definitions in the AOR and flags shipped terminal definitions which it has determined to be redundant. If any redundant definitions have been identified, 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 permitted to remain inactive before being flagged for deletion.

The **DSHIPINT** system initialization parameter determines the time interval between invocations of the CRMF transaction. CRMF examines 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 invokes CRMD to delete them.

Where useful

The CRMF/CRMD processing is 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.

Implementation

The maximum length of time for which a shipped terminal definition may remain idle before it can 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 can be adjusted. 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 it was last invoked, nor from the time of CICS startup.

Monitoring

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.

Limitations

The DSHIPINT value is the dominant factor in determining how long an idle shipped terminal definition survives before being deleted

After CRMF/CRMD processing has deleted a shipped terminal definition, the terminal definition must be reshipped when the terminal user next routes a transaction from the TOR to the AOR. DSHIPIDL values must not be set 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 reinstallation when the next transaction is routed.

Consider that a large value chosen for DSHIPINT influences the length of time that a shipped terminal definition survives. The period of time for which a shipped terminal definition remains idle before deletion is extended by an average of half of the DSHIPINT value. This occurs because a terminal, after it has exceeded the limit for idle terminals set by the DSHIPIDL parameter, must wait (for 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 is 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 survives before being deleted.

Recommendations

Do not assign too low a value 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 want to delete idle shipped terminal definitions incrementally or altogether. CRMF processing in itself causes negligible CPU overhead, so a low value for DSHIPINT may therefore be specified at little cost, if 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 CRMF identifies a considerable number of idle terminal definitions, so that a larger burst of CPU is required for the CRMD processing. To ensure that this type of processing occurs during periods of low activity in the CICS region, the INQUIRE/SET/PERFORM DELETSHIPPED commands are available to help you schedule when the CRMF transaction will be invoked.

Chapter 14. CICS VSAM and file control: Performance and tuning

This section describes performance tuning issues related to VSAM and file control.

VSAM tuning: General objectives

Tuning consists of providing a satisfactory level of service from a system at an acceptable cost. A satisfactory service for VSAM is likely to be obtained by providing adequate buffers to minimize physical I/O, and 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, including whether to use local shared resources or nonshared resources. If you compress CICS data sets to free storage and improve performance, you must do the compression while CICS is not running. To avoid shutting down CICS, use LIBRARY resources to easily take data sets offline for compression without affecting continuous availability. For details, see Using dynamic program LIBRARY resources in Developing applications.

A distinction is made between files and data sets:
- A *file* means a view of a data set as defined by an installed CICS file resource definition and a VSAM ACB.
- A *data set* means a VSAM sphere, including the base cluster with any associated alternate index paths.

Local shared resources (LSR) or nonshared resources (NSR)

You must decide for each file whether to use local shared resources (LSR) or nonshared resources (NSR) for its VSAM buffers and strings.

All files opened for access to a particular VSAM data set must typically use the same resource type.

Access to VSAM control intervals (CIs)

An important difference between LSR and NSR is in concurrent access to VSAM control intervals (CIs):
- 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.
- NSR allows multiple copies of a CI in storage. You can have one (and only one) string updating a CI and other strings reading different copies of the same CI.

However, 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 might, with LSR, encounter a deadlock as the second operation waits unsuccessfully for the first to complete.

The CICS monitoring facility provides performance data for the exclusive control wait time for each user task. The performance data field 426, FCXCWTT, in the DFHFILE group, shows the elapsed time in which the task waited for exclusive control of a VSAM control interval.

Size of control intervals (CIs)

The size of the data set CIs is not a parameter specified to CICS, and 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 the data CI is small, whereas sequential I/O is quicker when the data CI is large. With NSR files, it is possible to get a good compromise by using a small data CI 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. Set the data CI larger than the index CI. Thus, typical CI sizes for data are 4 KB to 12 KB, and for index, 1 KB to 2 KB.

In general, specify the size of the data CI for a file, but allow VSAM to select the appropriate index CI to match. An exception is if key compression turns out to be less efficient than VSAM expects it to be. In this case, VSAM might select too small an index CI size. You might find an unusually high rate of control area (CA) splits occurring with poor use of DASD space. If this problem is suspected, specify a larger index CI.

With LSR, there might be a benefit in standardizing the CI sizes, because this standardization allows more sharing of buffers between files and allows a lower total number of buffers. Conversely, there might be a benefit in giving a file unique CI sizes to prevent it from competing for buffers with other files that use the same pool.

Try to keep CI sizes at 512-bytes, 1 KB, 2 KB, or any multiple of 4 KB. Avoid unusual CI sizes like 26 KB or 30 KB. A CI size of 26 KB does not mean that physical block size is 26 KB; the physical block size is most likely to be 2 KB in this case because it is device-dependent.

Considerations for ESDS files

There are some special performance considerations when choosing a **STRINGS** value 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 suggested. 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.

Number of buffers

Some important differences exist between LSR and NSR in the way that VSAM allocates and shares the buffers.

The set of buffers of one size in an LSR pool is called a subpool. You use up to 255 separate LSR pools for file control files. You also must decide 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 is built and used for both data and index records. The number of buffers for each subpool is controlled by the DATA and INDEX parameters of the LSRPOOL definition. You can specify precise numbers or have CICS calculate the numbers.

NSR files or data sets have their own set of buffers and control blocks. Enough buffers must be provided for each file to support the concurrent accesses specified in the STRINGS parameter for the file. VSAM enforces this requirement for NSR. NSR is not supported for transactions that use transaction isolation. File control commands using NSR files are not threadsafe.

For more information, see "Number of buffers and strings for LSR and NSR" on page 186.

Number of strings

The next decision to make is the number of concurrent accesses to be supported for each file and for each LSR pool.

You must specify 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.

When deciding on 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.

For more information, see "Number of buffers and strings for LSR and NSR" on page 186.

Effects

LSR has significant advantages, by providing the following effects:
- 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.
- Better read integrity because there is only one copy of a CI in storage.
- Self-tuning because more buffers are allocated to busy files and frequently referenced index control intervals are kept in buffers.
- 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.

- Support for transaction isolation.

NSR can provide the following effects:
- Specific tuning in favor of a particular data set
- Better performance for sequential operations.

Suggestions

Use LSR for all VSAM data sets except where you have one of the following situations:
- A file is active but there is no opportunity for lookaside because, for example, the file is 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. 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. 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 also 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.

Limitations

All files with the same base data set, except read-only files with DSNSHARING(MODIFYREQS) specified in the file definition, must use either the same LSR pool, or all use NSR.

SERVREQ=REUSE files cannot use LSR.

Number of buffers and strings for LSR and NSR

The number of buffers and strings may affect your decision to use either LSR or NSR for each file.

Number of buffers for LSR and NSR

Some important differences exist between LSR and NSR in the way that VSAM allocates and shares the buffers:
- LSR

  Allowing CICS to calculate the LSR parameters is easy but it incurs additional processing to build the pool, when the first file that needs the LSR pool is opened. Consider the following factors if you allow CICS to calculate an LSR pool:
  - CICS must read the VSAM catalog for every file that is specified to use the pool.
  - The processing is increased if the data sets involved are migrated at the time that CICS performs the calculation. To enable CICS to read the VSAM catalog for each data set associated with the LSR pool, each data set must be recalled.

- Not only can a single recall cause a significant delay for the task that caused the recall, but it is a synchronous operation that delays other activities that CICS is running under the same TCB.

  You can avoid these delays by designing your SMS storage classes and migration policies to avoid CICS data sets being migrated. See the *DFSMShsm Storage Administration Reference* and the *DFSMShsm Storage Administration Guide* for information about setting data set migration criteria.

  CICS outputs an information message, DHFC0989, when a recall is necessary, advising you that the consequent delay is not an error situation.
- An LSR pool calculated by CICS cannot be fine-tuned by specifying actual sizes for each buffer.
- In LSR, there is no preallocation of buffers to strings, or to particular files or data sets. When VSAM must 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 might not have to issue the read. This buffer lookaside can reduce I/O significantly.
- 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.

  When changing the size of an LSR pool, refer to the CICS statistics before and after the change is made. These statistics show whether the proportion of VSAM reads satisfied by buffer lookaside is 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 benefit is a further reason for standardizing LSR data and index CI sizes, so that one subpool does not have a mix of index and data CIs in it.

  Because data and index buffers are specified separately with the LSRPOOL definition, there is no requirement to use CI size to differentiate between data and index values.

  Take care to include buffers of the correct size. If no buffers of the required size are present, VSAM uses the next larger buffer size.

• NSR
- Enough buffers must be provided for each file to support the concurrent accesses specified in the **STRINGS** parameter for the file. In fact, VSAM enforces this requirement for NSR.
- Specify the number of data and index buffers for NSR using the **DATABUFFERS** and **INDEXBUFFERS** parameters of the file definition. 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 **STRINGS** is sufficient. But when a KSDS is larger than this value, at least one extra index buffer must be specified so that at least the top-level index buffer is shared by all strings. Further index buffers reduce index I/O to some extent.
- Set **DATABUFFERS** to the minimum at STRINGS + 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.
- When the file is an alternate index path to a base, the same **INDEXBUFFERS** (if the base is a KSDS) and **DATABUFFERS** settings are used for alternate index and base buffers (see “CICS calculation of LSR pool parameters” on page 191). In NSR, the minimum number of data buffers is STRNO + 1, and the minimum index buffers (for KSDSs and alternate index paths) is STRNO. One data and one index buffer are preallocated to each string, and one data buffer is kept in

reserve for CA splits. If there are extra data buffers, these buffers are assigned to the first sequential operation; they can 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.

Note:

Always design and program transactions to avoid deadlocks. For further information, see *CICS Application Programming Guide*.

Number of strings

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. An access using an alternate index needs two strings (one to hold position on the alternate index 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. For each concurrent request, VSAM can reuse the n strings required for upgrade set processing because the upgrade set is updated serially.

A simple operation such as direct reading frees the string or strings immediately. However, a read for update, mass insert, or browse request retains the string or strings until a corresponding update, unlock, or end browse request is performed.

The interpretation of the **STRNO** parameter by CICS and by VSAM differs depending upon the context:
- The equivalent **STRINGS** parameter of the LSR pool definition (LSRPOOL) has the same meaning as the **STRNO** parameter 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 **STRINGS** value specified.
- The equivalent **STRINGS** parameter of the file definition has the same meaning as the **STRNO** parameter in the VSAM ACB for NSR files. That is, the actual number of concurrent outstanding VSAM requests that can be handled. When alternate index paths or upgrade sets are used, the actual number of strings that VSAM allocates to support these paths or upgrade sets can be greater than the **STRINGS** value specified.

For LSR, it is possible to specify the precise numbers of strings, or to have CICS calculate the numbers. The number specified in the LSR pool definition is the actual number of strings in the pool. If CICS calculates the number of strings, it derives the pool **STRINGS** from the RDO file definition. It interprets this pool, like with NSR, as the actual number of concurrent requests.

You must decide how many concurrent read, browse, update, mass insert requests, and so on, you must support.

If access to a file is read only with no browsing, there is no need to have many strings; just one might be sufficient. While a read operation only holds the VSAM string for the duration of the request, it might need to wait for the completion of an update operation on the same CI.

In general, where some browsing or updates are used, set **STRINGS** to 2 or 3 initially and check CICS file statistics regularly to see the proportion of wait-on strings encountered. Wait-on strings of up to 5% of file accesses would typically be considered acceptable. Do 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 STRINGS= specified in the file definition. For each LSR pool, CICS also prevents more requests being concurrently made to VSAM than can be handled by the strings in the pool. 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 monitoring facility provides performance data for the VSAM string wait time for each user task. The performance data field 427, FCVSWTT, in the DFHFILE group, shows the elapsed time in which the task waited for a VSAM string. The CICS LSR pool statistics give information about the number of strings, the number of requests that waited for strings, and the maximum number of strings that were active at one time.

When deciding on 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 can 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, 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 large key (primary or alternate) into an LSR pool with many strings might use excessive storage.

VSAM specifications for LSR

Define VSAM buffer allocations and string settings for LSR. Specify the resource percentile and the maximum key length for LSR.

Defining VSAM buffer allocations for LSR

For files that use local shared resources (LSR), the number of buffers to be used is not specified explicitly by file. The files share the buffers of appropriate sizes in the LSR pool. The number of buffers in the pool can either be specified explicitly using the **BUFFERS** parameter in the file definition on the CICS system definition data set (CSD), or you can leave it to CICS to calculate. For more information about the CSD, see Where resource definitions are held in the *CICS Resource Definition Guide*.

Use the **BUFFERS** parameter in CICS systems that use VSAM LSR files in CICS file control. It 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 optimum buffer allocation involves a trade-off 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. The optimum buffer allocation for LSR is likely to be less than the buffer allocation for the same files using NSR.

The effects of these parameters can be monitored through transaction response times and data set and paging I/O rates. The settings influence both file and LSRPOOL statistics. The CICS file statistics show data set activity of the VSAM data sets. The VSAM catalog and RMF can show data set activity, I/O contention, space usage, and control interval (CI) size.

Defining VSAM string settings for LSR

The **STRINGS** parameter is used to determine the number of strings and the number of concurrent operations possible against the LSR pool, assuming that there are buffers available. The **STRINGS** parameter can be used in CICS systems with VSAM data sets.

The number of strings is defined by the **STRNO** parameter in the file definition on the CSD, which limits the concurrent activity for that particular file.

The **STRINGS** parameter 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 **STRINGS** value 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 the cost of 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.

A maximum of 255 strings is allowed per pool. The effects of the **STRINGS** parameter can be seen in changes to the response times for each file entry. The CICS LSR pool statistics give information about the number of strings, the number of requests that waited for strings, and the maximum number of strings that were active at one time. The CICS performance data field 427, FCVSWTT, in the DFHFILE group, shows the elapsed time in which each user task waited for a VSAM 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 “File control statistics” on page 517), and one at the shared resource pool level (see “LSR pool statistics” on page 617).

Specifying the maximum key length for LSR

The **KEYLENGTH** parameter in the file definition in the CSD, or the **MAXKEYLENGTH** parameter in the LSR pool definition, specifies the size of the largest key to be used in an LSR pool. The **KEYLENGTH** parameter can be used in CICS systems with VSAM data sets. Specify the maximum key length explicitly using the **KEYLENGTH** parameter in the file definition on the CSD. Or leave it to CICS to determine the maximum key length from the VSAM catalog. For more information about the CSD, see Where resource definitions are held in the *CICS Resource Definition Guide*.

The **KEYLENGTH** parameter causes the placeholder control blocks to be built with space for the largest key that can be used with the LSR pool. Too small a specified **KEYLENGTH** prevents requests for files that have a longer key length. Set the key length so it is always as large as, or larger than, the largest key for files using the LSR pool.

Specifying the resource percentile for LSR

The **SHARELIMIT** parameter in the LSR pool definition specifies the percentage of the buffers and strings that CICS applies to the value that it calculates. The **SHARELIMIT** parameter can be used in CICS systems with VSAM data sets. The **SHARELIMIT** parameter is specified in the LSR pool definition. For more information, see Where resource definitions are held in the *CICS Resource Definition Guide*.

The **SHARELIMIT** parameter is ignored if both the **BUFFERS** and the **STRINGS** parameters are specified for the pool. **SHARELIMIT** can be applied only to files that are allocated at initialization of the LSR pool, when the first file in the pool is opened. Therefore, it is always wise to specify the decimal **STRINGS** and **BUFFERS** for an LSR pool.

CICS calculation of LSR pool parameters:

If you have not specified LSR parameters for a pool, CICS calculates the buffers and strings required for you.

To do this calculation, CICS scans all the installed file resource definitions for files specified to use the pool. For each file, it uses the following values:
- From the CICS file resource definitions:
  - The number of strings, as specified on the **STRINGS** parameter
- From the VSAM catalog:
  - The levels of index for each of these files
  - The control interval (CI) sizes
  - The keylengths for the base, the path (if it is accessed through an alternate index path), and upgrade set alternate index.

If you have specified only buffers or only strings, CICS performs the calculation for the buffers and strings you have not specified.

The following information helps you calculate the buffers required. A particular file might require more than one buffer size. For each file, CICS determines the buffer sizes required for the following components:
- The data component
- The index component, if it is a KSDS

- The data and index components for the alternate index, if it is an alternate index path
- The data and index components for each alternate index in the upgrade set, if any

The number of buffers for each file is calculated as follows:
- For data components for base and alternate index = (STRINGS= in the file resource definition entry) + 1
- For index components for base and alternate index = (STRINGS= in the file resource definition entry) + (the number of levels in the index) – 1
- For data and index components for each alternate index in the upgrade set, one buffer each

When this calculation has been done for all the files that use the pool, the total number of buffers for each size is further calculated as follows:
- The number is reduced to either 50% or the percentage specified in the **SHARELIMIT** in the LSRPOOL definition. The **SHARELIMIT** parameter takes precedence.
- If necessary, the number is increased to a minimum of three buffers.
- The number is rounded up to the nearest 4 KB boundary.

To calculate the number of strings, CICS determines the number of strings required to handle concurrent requests for each file as the sum of the following values:
- **STRINGS** parameter value for the base
- **STRINGS** parameter value for the alternate index (if it is an alternate index path)
- *n* strings if there is an upgrade set (where *n* is the number of members in the upgrade set).

Note: If the LSR pool is calculated by CICS and the data sets have been archived by hierarchical storage manager (HSM), when the first file that needs the LSR pool is opened, 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 region is started again, and remains until the data set has been opened.

When the strings have been accumulated for all files, the total number of buffers is further calculated as follows:
- The total is reduced to either 50% or the percentage specified in the **SHARELIMIT** parameter in the LSRPOOL definition. The **SHARELIMIT** parameter takes precedence.
- The total is reduced to 255 (the maximum number of strings allowed for a pool by VSAM).
- The total is increased to the largest specified **STRINGS** value for a particular file.

The parameters calculated by CICS are shown in the CICS statistics.

Switching data sets from RLS mode to LSR mode

There might be occasions when you must switch a data set from RLS mode to non-RLS mode (for example, to read-only LSR mode during a batch update). This switch could lead to the LSR pools that are not explicitly defined, and which CICS builds using default values, not having sufficient 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 includes 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 that is being built using values that CICS calculates, specify `RLSTOLSR=YES` for this system initialization parameter (`RLSTOLSR=NO` is the default)

See RLSTOLSR in the *CICS System Definition Guide* for more information about this parameter.

Data set name sharing

Data set name (DSN) sharing is the default for all VSAM data sets. It is specified as MACRF=DSN in the VSAM ACB. 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 does VSAM realize that it is processing the same data set.

This single structure offers the following benefits:
- It provides VSAM update integrity for multiple access control blocks (ACB) updating one VSAM data set.
- It allows the use of VSAM share options 1 or 2, while still permitting multiple update blocks within the CICS region.
- It 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 (*READ=YES* or *BROWSE=YES*) and with *DSNSHARING(MODIFYREQS)* in the file resource definition. CICS provides this option so that a file (represented by an installed file resource definition) can be isolated from other users of that same data set in a different LSR pool or in NSR by suppressing DSN sharing. CICS ignores this parameter 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 **NSRGROUP** parameter is associated with DSN sharing. It is used to group file resource definitions that are to refer to the same VSAM base data set. NSRGROUP=*name* does not affect 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, with 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 by adding the STRINGS values in all the files that share the same **NSRGROUP** parameter.

If you do not provide the **NSRGROUP** parameter, the VSAM control block structure can be built with insufficient strings for later processing. Avoid this structure 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.

Alternate index considerations

For each alternate index defined with the **UPGRADE** attribute, VSAM upgrades the alternate index automatically when the base cluster is updated.

For NSR, VSAM uses a special set of buffers associated with the base cluster. This set consists of two data buffers and one index buffer, which are used serially for each alternate index 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.

Take care when specifying to VSAM that an alternate index is 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 alternate index in the upgrade set. This update involves extra processing and extra I/O operations.

Situations that cause extra physical I/O

Some situations that can lead to many physical I/O operations, thus affecting both response times and associated processor path lengths, are as follows:
- When a KSDS is defined with SHROPT of 4, all direct reads cause a refresh of both index and data buffers (to ensure the latest copy).
- Any sequence leading to CICS issuing ENDREQ invalidates all data buffers associated with the operation. This situation might 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 sync point 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 parameters

Select free space parameters with care, because these parameters can help reduce the number of control interval (CI) and control area (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, allow VSAM 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 or 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 specifications for NSR

Defining VSAM string settings for NSR and defining VSAM buffer allocations for NSR.

Defining VSAM buffer allocations for NSR

For files using nonshared resources (NSR), the **INDEXBUFFERS** and **DATABUFFERS** parameters define VSAM index buffers and data buffers.

The **INDEXBUFFERS** and **DATABUFFERS** parameters are defined in the file definition on the CSD. They correspond exactly to VSAM ACB parameters: **INDEXBUFFERS** is the number of index buffers, **DATABUFFERS** is the number of data buffers.

- Effects

  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 control area (CA) splits. Providing extra buffers for high-level index records can reduce physical I/O operations.

  Buffer allocations above the 16 MB line represent a significant part of the virtual storage requirement of most CICS systems.

- Limitations

  These parameters 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 value is automatically reduced to 255. Never override VSAM strings and buffers by specifying the **AMP** attribute on the DD statement.

- Limitations

  The effects of these parameters 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 control interval (CI) size.

Defining VSAM string settings for NSR

The **STRINGS** parameter is used to determine the number of concurrent operations possible against the file, and against the VSAM base cluster to which the file relates.

Use the **STRINGS** parameter in CICS systems that use VSAM NSR files in CICS file control.

The number of strings is defined by the **STRINGS** parameter in the CICS file definition on the CSD. It corresponds to the VSAM parameter 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** value for the ACB.

- Effects

  The **STRINGS** parameter 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 **STRINGS** value in the VSAM ACB.
  - It is used, with the **BASE** parameter, to calculate the VSAM **BSTRNO** value.
  - 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 the cost of 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 16 MB line.

- Limitations

  A maximum of 255 strings can be used as the **STRNO** or **BSTRNO** values in the ACB.

- Monitoring

  The effects of the **STRINGS** parameter can be seen in changes to response times. The CICS performance data field 427, FCVSWTT, in the DFHFILE group, shows the elapsed time in which each user task waited for a VSAM string. The CICS LSR pool statistics give information about the number of strings, the number of requests that waited for strings, and the maximum number of strings that were active at one time. RMF can show I/O contention in the DASD subsystem.

Using VSAM subtasking

The optional concurrent (CO) mode TCB is used for processes that can safely run in parallel with other CICS activity such as VSAM requests. The SIT keyword **SUBTSKS** has been defined to have numeric values (0 and 1) to specify whether there is to be a CO TCB. The system initialization parameter, **SUBTSKS=1**, defines that subtasking is to be used.

Subtasking is useful with CICS systems that use VSAM.

Only use subtasking 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.

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 control interval (CI) or control area (CA) splitting in NSR pools, causes only the additional TCB to stop. This effect might allow more throughput in a region that has many CA splits in its file, but has to be assessed cautiously regarding the extra processing associated with using the subtask.

When the **SUBTSKS=1** system initialization parameter has been specified, the following subtasks effects are seen:

- All non-RLS 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.

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 (UP). Use it only for a region that reaches the maximum capacity of one processor in a complex system that has spare processor capacity, or has NSR files that undergo frequent CI or 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 might increase or decrease because of conflict between subtasking processing and better use of multiprocessors. Task-related DSA occupancy increases or decreases proportionately.

Suggestions

Specify **SUBTSKS=1** only when the CICS system is run on an MVS image with two or more processors, and the peak processor utilization due to the CICS main TCB in a region exceeds 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 used to perform the I/O scheduling activity for VSAM data sets, auxiliary temporary storage, and intrapartition transient data.

The maximum system throughput of this 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 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.

Consider subtasking 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).

VSAM subtasking for threadsafe applications using local VSAM LSR or RLS, with FCQRONLY=NO set in the SIT, is not normally recommended. Performance benefits are greater for threadsafe file control applications, by using multiple L8 or L9 TCBs.

Monitoring

CICS dispatcher domain statistics include information about the modes of TCB listed in “Dispatcher TCB Modes report” on page 818.

CMF data and CICS trace are fully available.

Using 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 16 MB line. Therefore, they can provide a substantial performance benefit by reducing DASD I/O and path length resources. The path length to retrieve a record from a data table is shorter than the path length to retrieve a record that is already in a VSAM buffer.

You can define data tables using either the DEFINE FILE command of the CED*x* transaction or the DFHCSDUP utility program. See *CICS Resource Definition Guide* for more information.

Effects

Using data tables has the following 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 (CMTs).
- Reductions in DASD I/O for CMTs are dependent on the READ/WRITE ratio. This ratio is the number of READ to WRITE calls that are experienced on the source data set, before the data table implementation. These reductions also depend on the data table READ-hit ratio: the number of READ calls 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 reduction 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 CMTs, CICS ensures the synchronization of source data set and data table changes. When a file is recoverable, the necessary synchronization is already implemented by the existing record locking. When the file is unrecoverable, there is no CICS record locking and the note string position (NSP) mechanism is used instead for all update requests. This action might have a small performance impact of additional VSAM ENDREQ requests in some instances.

Suggestions

Data tables are defined by two RDO parameters of the file definition, **TABLE** and **MAXNUMRECS** . No other changes are required.

Begin by selecting only one or two candidates. You might want to start with a CMT to simplify recovery considerations.

Select a CMT with a high READ to WRITE ratio. This information can be found in the CICS LSRPOOL statistics (see topicpage “LSR pool statistics” on page 617) by running a VSAM LISTCAT job.

Use READ INTO, because READ SET incurs slightly more internal processing.

Monitor your real storage consumption. If your system is already real-storage constrained, having large data tables could increase your page-in rates, and in turn could adversely affect CICS system performance. Use your normal performance tools such as RMF to look at real storage and paging rates.

Select files that have a high proportion of full keyed direct reads as CMT 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 both modify and select records. This action could allow the user-maintained data table to contain only the information relevant to the application.

If storage isolation is specified, you must allow for the extra storage needed by the data tables to prevent CICS incurring increased paging.

Try to avoid the situation where two open files, one defined as a CMT and the other as a VSAM file, refer to the same underlying VSAM sphere (for example, both refer to the same data set name). In this situation, the VSAM file is treated almost as if it were a CMT, meaning that it gets both the advantages and disadvantages of a CMT. The advantage is much faster read and browse processing from the table created for the other file.

The disadvantages for the performance of the VSAM file are as follows:
- Updates must update both the file and the table.
- If the VSAM file refers to a path rather than to the base (that is, it uses alternate keys) it loses the advantage of fast reads.
- Requests for the VSAM file are always switched to the QR task control block (TCB) and are not processed on an open TCB.

Monitoring

Performance statistics are gathered to assess the effectiveness of the data table. They are in addition to the statistics 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 they were being added to the table either during loading or through the API
- The number of attempts to add a record that failed due to the table being full (already at its maximum number of records)
- The number of attempts to update table records through rewrite requests.
- The number of attempts to delete records from the table
- The highest value that the number of records in the table has reached since it was last opened.

There are circumstances in which apparent discrepancies in the statistics might be seen, caused, for example, by the existence of in-flight updates.

Using coupling facility data tables

The API used to store and retrieve the data from a coupling facility data table (CFDT) is based on the file control API used for user-maintained data tables.

A CFDT is similar in many ways to a shared user-maintained data table. For information about shared data tables, see Introduction to shared data tables.

A CFDT is defined to a CICS region using a FILE definition with the following parameters:
- **TABLE(CF)**
- **MAXNUMRECS(NOLIMIT***number*(1 through 99999999))
- **CFDTPOOL**(*pool_name*)
- **TABLENAME**(*name*)

- **UPDATEMODEL(CONTENTION|LOCKING)**
- **LOAD(NO|YES)**

MAXNUMRECS specifies the maximum number of records that CFDT can hold.

The first CICS region to open the CFDT determines the attributes for the file. Once opened successfully, these attributes remain associated with the CFDT through the data in the coupling facility list structure. Unless this table or coupling facility list structure is deleted or altered by a CFDT server operator command, the attributes persist even after CICS and CFDT server restarts. Other CICS regions attempting to open the CFDT must have a consistent definition of the CFDT, for example using the same update model.

The CFDT server controls the coupling facility list structure and the data tables held in this structure. The parameters documented in Coupling facility data table server parameters describe how initial structure size, structure element size, and entry-to-element ratio can be specified.

The data, unlike a UMT, is not kept in a dataspace in an MVS image and controlled by a CICS region, but kept in a coupling facility list structure. Control is shared between CFDT server regions. A CICS region requesting access to a CFDT communicates with a CFDT server region running in the same MVS image, using the MVS authorized cross-memory (AXM) server environment. The same technique is used by CICS temporary storage servers.

CFDTs are useful for informal shared data. Uses could include a sysplex-wide shared scratchpad, look-up tables of telephone numbers, and creating a subset of customers from a customer list. Compared with existing methods of sharing data of this kind, such as shared data tables, shared temporary storage or RLS files, CFDTs offer some distinct advantages:

- If the data is frequently accessed for modification, CFDT provides superior performance compared with function-shipped UMT requests, or using an RLS file
- CFDT-held data can be recoverable within a CICS transaction. Recovery of the structure is not supported, but the CFDT record is recoverable in the event of a unit of work failure, a CICS region failure, a CFDT server failure, or an MVS failure (that is, updates made by units of work that were in-flight at the time of the failure are backed out). Such recoverability is not provided by shared temporary storage.

Locking model and contention model

There are two models of coupling facility data table, a contention model or locking model.

Locking model. Records held in a coupling facility list structure are marked as locked by updating the adjunct area associated with the coupling facility list structure element that holds the data. Locking a record requires an additional coupling facility access to set the lock, having determined on the first access that the data was not already locked.

If, however, there is an update conflict, a number of extra coupling facility accesses are needed, as described in the following sequence of events:

1. The request that encounters lock contention is initially rejected.

2. The requester modifies the locked record adjunct area to express an interest in it. This area is a second extra coupling facility access for the lock waiter.
3. The lock owner has the update rejected because the record adjunct area has been modified, requiring the CICS region to read and try the update again. This results in two extra coupling facility accesses.
4. The lock owner sends a lock release notification message. If the lock was requested by a different server, this results in a coupling facility access to write a notification message to the other server and a coupling facility access to read it on the other side.

Contention model. The contention update model uses the entry version number to track changes. The entry version number is changed each time the record is updated. This change allows an update request to check that the record has not been altered since its copy of the record was acquired.

When an update conflict occurs, additional coupling facility accesses are needed:
- The request that detects that the record has changed is initially rejected and a CHANGED response is sent.
- The application receiving the response has to decide whether to try the request again.

Using the contention model, an exception condition (CHANGED) notifies an application that a rewrite following a read for update, or a delete following a read for update, needs to be tried again because the copy of the record in the table has been updated by another task before the rewrite or delete could be performed. The contention model does not lock a record, but uses the version number of the table entry for the record to check that it has not been altered. If the version of this record on rewrite or delete is not the same as when the original read for update was performed, the CHANGED condition is returned.

The locking model causes records to be locked following a read for update request so that multiple updates cannot occur.

A contention model CFDT is unrecoverable. A locking model CFDT can be recoverable or unrecoverable. For an unrecoverable locking model, CFDT locks are held until a read for update sequence is completed by a rewrite, a delete or an unlock request, but not until the next syncpoint. Changes are not backed out if a unit of work fails. In the recoverable case, locks are held until syncpoint, and the CFDT record is recoverable in the event of a unit of work failure, CICS region failure, CFDT server failure, or MVS failure.

The relative cost of using update models and recovery is related to the amount of coupling facility accesses needed to support a request. Contention requires the least number of accesses, but if the data is changed, additional programming and coupling facility accesses would be needed to handle this condition. Locking requires more coupling facility accesses, but does mean that a request does not need to be tried again, whereas repeat tries can be required when using the contention model. Recovery also requires further coupling facility accesses, because the recovery data is kept in the coupling facility list structure.

The following table shows the amount of coupling facility accesses needed to support the CFDT request types by update model.

Table 15. Coupling facility access by request type and update model

Request description	Contention	Locking	Recoverable
Open, Close	3	3	6
Read, Point	1	1	1
Write new record	1	1	2
Read for Update	1	2	2
Unlock	0	1	1
Rewrite	1	1	3
Delete	1	1	2
Delete by key	1	2	3
Syncpoint	0	0	3
Lock WAIT	0	2	2
Lock POST	0	2	2
Cross-system POST	0	2 per waiting server	2 per waiting server

For a description of how to define a coupling facility data table (CFDT), and start a coupling facility data table server, see Defining a coupling facility data table pool in the *CICS System Definition Guide*.

Effects

In a test that compared the use of a CFDT with a function-shipped UMT between 2 CICS regions running on different MVS members of a sysplex, it was found that overall CPU utilization was reduced by over 40% by using CFDTs. Some general observations that might be useful are as follows:

- Access to CFDT records of 4094 bytes or less (4096 K or 4 K including 2 bytes of prefix data) are handled as synchronous coupling facility requests by the CFDT server. Requests for records of greater than 4 K bytes are made asynchronously. These asynchronous accesses cost a little more in CPU usage and response time. In a benchmark test comparing the same transaction rates (337 per second) but different record sizes, the less than 4 K CFDT workload took 41.7% less CPU than the UMT equivalent. The greater than 4 K CFDT workload took 41.1% less CPU with no measurable degradation of response time.
- Using the contention model requires the least coupling facility accesses but because the CHANGED condition needs to be handled and might need to be tried again, maximum benefit is derived when there are few CHANGED conditions. These occurrences are reported in the CICS statistics which follow.
- If the CFDT records are 63 bytes or less in length, the record data is stored in the entry adjunct area of the coupling facility list structure, which gives improved performance when using the contention update mode.
- Using the locking model with recovery is the most costly mode of CFDT operation. Not only does this require more coupling facility accesses, but the CFDT server is also acting as a resource manager, coordinating the committal of updates with the requesting CICS region. In a benchmark test involving the READ/UPDATE and REWRITE of CFDT records at a transaction rate of 168 per second, there was no significant difference in CPU utilization between transactions using contention and locking CFDTs. However, if the CFDT was defined as recoverable, the CPU utilization of the same transactions increased by approximately 15%.

Suggestions

Choose an appropriate use of a CFDT. For example, for cross-system, recoverable scratchpad storage, where shared TS does not give the required functionality, or VSAM RLS incurs too much processing.

A large file requires a large amount of coupling facility storage to contain it. Smaller files are better CFDT candidates (unless your application is written to control the number of records held in a CFDT).

The additional cost of using a locking model compared with a contention model is not great. Considering that using the contention model might need application changes if you are using an existing program, locking is probably the best choice of update model for your CFDT. If coupling facility accesses are critical to you, they are minimized by the contention model.

Recovery costs slightly more in CPU usage and in coupling facility utilization.

Allow for expansion when sizing the CFDT. The amount of coupling facility storage a structure occupies can be increased dynamically up to the maximum defined in the associated coupling facility resource management (CFRM) policy with a **SETXCF ALTER** command. The **MAXTABLES** value defined to the CFDT server allows for expansion. Therefore, consider setting it to a value higher than your initial requirements. If a CFDT does become full, its capacity can be increased using the CFDT operator command `SET TABLE=name,MAXRECS=n`.

Monitor the utilization of the CFDT regularly both through CICS and CFDT statistics and RMF. Check that the size of the structure is reasonable for the amount of data it contains. A maximum used of 80% is a reasonable target. Define a maximum coupling facility list structure size in the CFRM policy definition greater than the initial allocation size specified by the **POOLSIZE** parameter in the CFDT server startup parameters. This setting enables you to enlarge the structure dynamically with a **SETXCF ALTER** command if the structure does fill, in extraordinary circumstances.

Ensure that the AXMPGANY storage pool is large enough. This pool can be increased by increasing the REGION size for the CFDT server. Insufficient AXMPGANY storage might lead to 80A abends in the CFDT server.

Monitoring

Both CICS and the CFDT server produce statistics records. These records are described in “Coupling facility data tables server statistics” on page 459.

The CICS file statistics report the various requests by type issued against each CFDT. They also report if the CFDT becomes full, the highest number of records held and a Changed Response/Lock Wait count. This last item can be used to determine for a contention CFDT how many times the CHANGED condition was returned. For a locking CFDT, this count reports how many times requests were made to wait because the requested record was already locked.

For more information, see “File control statistics” on page 517.

Coupling facility data table statistics

The coupling facility data table (CFDT) server reports comprehensive statistics on both the coupling facility list structure it uses and the data tables it supports. It also reports on the storage that is used within the CFDT region by its AXM routines (the AXMPGLOW and AXMPGANY areas). This data can be written to SMF and can also be produced automatically at regular intervals, or by operator commands to the job log of the CFDT server.

The CFDT statistics are calculated from information that is returned by recent coupling facility requests. If the relevant information was not accessed recently by the current server, the statistics are not necessarily accurate. The number of tables and the number of lists are updated each time that the server opens or closes a table, but at other times they might not be updated. The element and entry counts are updated on successful completion of most types of coupling facility access request.

The following code is an example of coupling facility statistics that are produced by a CFDT server:

```
DFHCF0432I Table pool statistics for coupling facility list structure DFH
CFLS_PERFCFT2:
 Structure:    Size  Max size Elem size   Tables:   Current   Highest
            12288K    30208K       256                    4         4
 Lists:       Total    In use  Max used   Control      Data
                137        41        41        37         4
               100%       30%       30%       27%        3%
 Entries:     Total    In use  Max used      Free  Min free   Reserve
               3837      2010      2010      1827      1827       191
               100%       52%       52%       48%       48%        5%
 Elements:    Total    In use  Max used      Free  Min free   Reserve
              38691     12434     12434     26257     26257      1934
               100%       32%       32%       68%       68%        5%
```

This example shows the amount of space that is currently used in a coupling facility list structure (Size) and the maximum size (Max size) defined for the structure. The structure size can be increased by using a **SETXCF ALTER** command. The number of lists that are defined is determined by the **MAXTABLES** parameter for the CFDT server. In this example, the structure can support up to 100 data tables (and 37 lists for control information).

Each list entry comprises a fixed-length section for entry controls and a variable number of data elements. The size of these elements is fixed when the structure is first allocated in the coupling facility, and is specified to the CFDT server by the **ELEMSIZE** parameter. The allocation of coupling facility space between entry controls and elements is altered automatically and dynamically by the CFDT server to improve space utilization if necessary.

The reserve space is used to ensure that rewrites and server internal operations can still function if a structure fills with user data.

The amount of storage that is used with the CFDT region to support AXM requests is also reported. For example:

```
AXMPG0004I Usage statistics for storage page pool AXMPGANY:
     Size    In Use  Max Used      Free  Min Free
   30852K      636K      672K    30216K    30180K
     100%        2%        2%       98%       98%
               Gets     Frees   Retries     Fails
               3122      3098         0         0
AXMPG0004I Usage statistics for storage page pool AXMPGLOW:
```

```
Size    In Use  Max Used      Free  Min Free
440K       12K       12K      428K      428K
100%        3%        3%       97%       97%
          Gets     Frees   Retries     Fails
             3         0         0         0
```

The CFDT server uses storage in its own region for AXMPGANY and AXMPGLOW storage pools. AXMPGANY accounts for most of the available storage above 16 MB in the CFDT region. The AXMPGLOW refers to 24 bit addressed storage (below 16 MB) and accounts for only 5% of this storage in the CFDT region. The CFDT server has a small requirement for such storage.

Local shared resources (LSR) or nonshared resources (NSR)

You must decide for each file whether to use local shared resources (LSR) or nonshared resources (NSR) for its VSAM buffers and strings.

All files opened for access to a particular VSAM data set must typically use the same resource type.

Access to VSAM control intervals (CIs)

An important difference between LSR and NSR is in concurrent access to VSAM control intervals (CIs):

- 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.
- NSR allows multiple copies of a CI in storage. You can have one (and only one) string updating a CI and other strings reading different copies of the same CI.

However, 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 might, with LSR, encounter a deadlock as the second operation waits unsuccessfully for the first to complete.

Size of control intervals (CIs)

The size of the data set CIs is not a parameter specified to CICS, and 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 the data CI is small, whereas sequential I/O is quicker when the data CI is large. With NSR files, it is possible to get a good compromise by using a small data CI 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. Set the data CI larger than the index CI. Thus, typical CI sizes for data are 4 KB to 12 KB, and for index, 1 KB to 2 KB.

In general, specify the size of the data CI for a file, but allow VSAM to select the appropriate index CI to match. An exception is if key compression turns out to be less efficient than VSAM expects it to be. In this case, VSAM might select too small an index CI size. You might find an unusually high rate of control area (CA) splits occurring with poor use of DASD space. If this problem is suspected, specify a larger index CI.

With LSR, there might be a benefit in standardizing the CI sizes, because this standardization allows more sharing of buffers between files and allows a smaller total number of buffers. Conversely, there might be a benefit in giving a file unique CI sizes to prevent it from competing for buffers with other files that use the same pool.

Try to keep CI sizes at 512 bytes, 1 KB, 2 KB, or any multiple of 4 KB. Avoid unusual CI sizes like 26 KB or 30 KB. A CI size of 26 KB does not mean that physical block size is 26 KB; the physical block size is most likely to be 2 KB in this case because it is device-dependent.

Number of buffers for LSR and NSR

Some important differences exist between LSR and NSR in the way that VSAM allocates and shares the buffers:

- LSR

  The set of buffers of one size in an LSR pool is called a *subpool*. You use up to 255 separate LSR pools for file control files. You also must decide 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 is built and used for both data and index records. The number of buffers for each subpool is controlled by the **DATA** and **INDEX** parameters of the LSRPOOL definition. You can specify precise numbers or have CICS calculate the numbers.

  Allowing CICS to calculate the LSR parameters is easy but it incurs additional processing to build the pool, when the first file that needs the LSR pool is opened. Consider the following factors if you allow CICS to calculate an LSR pool:

  – CICS must read the VSAM catalog for every file that is specified to use the pool.
  – The processing is increased if the data sets involved are migrated at the time that CICS performs the calculation. To enable CICS to read the VSAM catalog for each data set associated with the LSR pool, each data set must be recalled.
  – Not only can a single recall cause a significant delay for the task that caused the recall, but it is a synchronous operation that delays other activities that CICS is running under the same TCB.

    You can avoid these delays by designing your SMS storage classes and migration policies to avoid CICS data sets being migrated. See the *DFSMShsm Storage Administration Reference* and the *DFSMShsm Storage Administration Guide* for information about setting data set migration criteria.

    CICS outputs an information message, DHFC0989, when a recall is necessary, advising you that the consequent delay is not an error situation.
  – An LSR pool calculated by CICS cannot be fine-tuned by specifying actual sizes for each buffer.
  – In LSR, there is no preallocation of buffers to strings, or to particular files or data sets. When VSAM must 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 might not have to issue the read. This buffer lookaside can reduce I/O significantly.

  - 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.

    When changing the size of an LSR pool, refer to the CICS statistics before and after the change is made. These statistics show whether the proportion of VSAM reads satisfied by buffer lookaside is 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 benefit is a further reason for standardizing LSR data and index CI sizes, so that one subpool does not have a mix of index and data CIs in it.

    Because data and index buffers are specified separately with the LSRPOOL definition, there is no requirement to use CI size to differentiate between data and index values.

    Take care to include buffers of the correct size. If no buffers of the required size are present, VSAM uses the next larger buffer size.

- NSR
  - Enough buffers must be provided for each file to support the concurrent accesses specified in the **STRINGS** parameter for the file. In fact, VSAM enforces this requirement for NSR.
  - Specify the number of data and index buffers for NSR using the **DATABUFFERS** and **INDEXBUFFERS** parameters of the file definition. 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 **STRINGS** is sufficient. But when a KSDS is larger than this value, at least one extra index buffer must be specified so that at least the top-level index buffer is shared by all strings. Further index buffers reduce index I/O to some extent.
  - Set **DATABUFFERS** to the minimum at STRINGS + 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.
  - When the file is an alternate index path to a base, the same **INDEXBUFFERS** (if the base is a KSDS) and **DATABUFFERS** settings are used for alternate index and base buffers (see “CICS calculation of LSR pool parameters” on page 191). In NSR, the minimum number of data buffers is STRNO + 1, and the minimum index buffers (for KSDSs and alternate index paths) is STRNO. One data and one index buffer are preallocated to each string, and one data buffer is kept in reserve for CA splits. If there are extra data buffers, these buffers are assigned to the first sequential operation; they can 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.
  - NSR files or data sets have their own set of buffers and control blocks.

Note: NSR is not supported for transactions that use transaction isolation. File control commands using NSR files are not threadsafe.

Always design and program transactions to avoid deadlocks. For further information, see *CICS Application Programming Guide*.

Number of strings

The next decision to make is the number of concurrent accesses to be supported for each file and for each LSR pool.

You must specify 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. An access using an alternate index needs two strings (one to hold position on the alternate index 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. For each concurrent request, VSAM can reuse the n strings required for upgrade set processing because the upgrade set is updated serially.

A simple operation such as direct reading frees the string or strings immediately. However, a read for update, mass insert, or browse request retains the string or strings until a corresponding update, unlock, or end browse request is performed.

The interpretation of the **STRNO** parameter by CICS and by VSAM differs depending upon the context:
- The equivalent **STRINGS** parameter of the LSR pool definition (LSRPOOL) has the same meaning as the **STRNO** parameter 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 **STRINGS** value specified.
- The equivalent **STRINGS** parameter of the file definition has the same meaning as the **STRNO** parameter in the VSAM ACB for NSR files. That is, the actual number of concurrent outstanding VSAM requests that can be handled. When alternate index paths or upgrade sets are used, the actual number of strings that VSAM allocates to support these paths or upgrade sets can be greater than the **STRINGS** value specified.

For LSR, it is possible to specify the precise numbers of strings, or to have CICS calculate the numbers. The number specified in the LSR pool definition is the actual number of strings in the pool. If CICS calculates the number of strings, it derives the pool **STRINGS** from the RDO file definition. It interprets this pool, like with NSR, as the actual number of concurrent requests.

You must decide how many concurrent read, browse, update, mass insert requests, and so on, you must support.

If access to a file is read only with no browsing, there is no need to have many strings; just one might be sufficient. While a read operation only holds the VSAM string for the duration of the request, it might need to wait for the completion of an update operation on the same CI.

In general, where some browsing or updates are used, set **STRINGS** to 2 or 3 initially and check CICS file statistics regularly to see the proportion of wait-on strings encountered. Wait-on strings of up to 5% of file accesses would typically be considered acceptable. Do 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 STRINGS= specified in the file definition. For each LSR pool, CICS also prevents more requests being concurrently made to VSAM than can be handled by the strings in the pool. 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 on 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 can 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, 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 large key (primary or alternate) into an LSR pool with many strings might use excessive storage.

Considerations for ESDS files

There are some special performance considerations when choosing a **STRINGS** value 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 suggested. 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.

Effects

LSR has significant advantages, by providing the following effects:
- 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.
- Better read integrity because there is only one copy of a CI in storage.
- Self-tuning because more buffers are allocated to busy files and frequently referenced index control intervals are kept in buffers.

- 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.
- Support for transaction isolation.

NSR can provide the following effects:
- Specific tuning in favor of a particular data set
- Better performance for sequential operations.

Suggestions

Use LSR for all VSAM data sets except where you have one of the following situations:
- A file is active but there is no opportunity for lookaside because, for example, the file is 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. 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. 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 also 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.

Limitations

All files with the same base data set, except read-only files with DSNSHARING(MODIFYREQS) specified in the file definition, must use either the same LSR pool, or all use NSR.

SERVREQ=REUSE files cannot use LSR.

Coupling facility data tables

The CPU instruction data provided here was obtained using a 9672-R55 system.

Two tables are provided:
- The first for record lengths that result in synchronous coupling facility accesses (less than 4K)
- The second for record lengths that result in asynchronous coupling facility accesses (greater than 4K).

Note that asynchronous requests take more CPU time to process. The response times are also slightly longer than for synchronous requests. CPU instructions per API call for record lengths less than 4 K are as follows:

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	11.8	11.8	11.8
READ/UPDATE	12.0	22.2	22.4
REWRITE	19.5	24.0	33.0
WRITE	8.0	8.0	13.0
DELETE	7.0	11.0	16.5

CPU instructions per API call for record lengths greater than 4 K are as follows:

API CALL	CONTENTION	LOCKING	RECOVERABLE
READ	15.3	15.3	15.3
READ/UPDATE	15.0	25.7	25.9
REWRITE	23.0	27.5	36.5
WRITE	11.5	11.5	16.5
DELETE	10.5	14.5	20.0

Using VSAM record-level sharing

VSAM record-level sharing (RLS) is a VSAM data set access mode, introduced in DFSMS, and supported by CICS. RLS enables VSAM data to be shared, with full update capability, between many applications running in many CICS regions. With RLS, CICS regions that share VSAM data sets can reside in one or more MVS images within an MVS sysplex.

RLS also provides some benefits when data sets are shared between CICS regions and batch jobs.

RLS involves the use of the following components:

- **A VSAM server, subsystem SMSVSAM**. This subsystem runs in its own address space to provide the RLS support required by CICS application owning regions (AORs) and batch jobs, within each MVS image in a Parallel Sysplex® environment.

  The CICS interface with SMSVSAM is through an access control block (ACB), and CICS registers with this ACB to open the connection. Unlike the DB2 and DBCTL database manager subsystems, which require user action to open the connections, if you specify **RLS=YES** as a system initialization parameter, CICS registers with the SMSVSAM control ACB automatically during CICS initialization.

  A CICS region must open the control ACB to register with SMSVSAM before it can open any file ACB in RLS mode. Each normal file ACB remains the interface for file access requests.
- **Sharing-control data sets**. VSAM requires a number of these data sets for RLS control. The VSAM sharing control data sets are logically partitioned, linear data sets. They can be defined with secondary extents, but all the extents for each data set must be on the same volume.

  Define at least three sharing-control data sets. VSAM requires two active data sets for use in duplexing mode, and a third data set as a spare in case one of the active data sets fails.

  For more information about sharing-control data sets, and for a JCL example to define them, see *z/OS DFSMS Storage Administration Reference*.

- **Common buffer pools and control blocks.** For data sets accessed in non-RLS mode, VSAM control blocks and buffers (local shared resources (LSR) pools) are located in each CICS address space. They are thus not available to batch programs, and not even to another CICS region.

  With RLS, all the control blocks and buffers are allocated in an associated data space of the SMSVSAM server. This structure provides one large buffer pool for each MVS image, which can be shared by all CICS regions that are connected to the SMSVSAM server, and also by batch programs. Buffers in this data space are created and freed automatically.

  DFSMS provides the **RLS_MAX_POOL_SIZE** parameter that you can specify in the IGDSMS*xx* SYS1.PARMLIB member. There are no other tuning parameters for RLS as there are with LSR pools. Management of the RLS buffers is fully automatic.

Using RLS with entry-sequenced data sets (ESDS) can have a negative effect on the performance and availability of the data set when you are adding records. The following issues have been identified:

- When new records are added to the end of an ESDS in RLS access mode, the acquisition of locks on the various calls required to VSAM to satisfy the request might cause long response times for the operation.
- If a CICS region fails while writing to an ESDS, the data set might be locked until the CICS region is restarted.

For these reasons, do not use RLS with entry-sequenced data sets.

To use RLS access mode with CICS files, do the following tasks:

1. Define the required sharing control data sets.
2. Specify the **RLS_MAX_POOL_SIZE** parameter in the IGDSMS*xx* SYS1.PARMLIB member.
3. Ensure that the SMSVSAM server is started in the MVS image for which you want RLS support.
4. Specify the system initialization parameter **RLS=YES**. This parameter enables CICS to register automatically with the SMSVSAM server by opening the control ACB during CICS initialization. RLS support cannot be enabled dynamically later if you start CICS with RLS=NO.
5. Ensure that the data sets you plan to use in RLS-access mode are defined, using Access Method Services (AMS), with the required recovery attributes using the **LOG** and **LOGSTREAMID** parameters on the IDCAMS DEFINE statements. If you use an existing data set that was defined without these attributes, redefine the data set with these attributes specified.
6. Specify **RLSACCESS(YES)** on the file resource definition.

CICS can use three different modes to access a VSAM file. These are non-shared resources (NSR) mode, local shared resources (LSR) mode, and record-level sharing (RLS) mode. (CICS does not support VSAM global shared resources (GSR) access mode.) The mode of access is not a property of the data set itself, it is a property of the way that the data set is opened. This means that a given data set can be opened by a user in NSR mode at one time, and RLS mode at another. The term non-RLS mode is used as a generic term to refer to the NSR or LSR access modes supported by CICS. Mixed-mode operation means a data set that is opened in RLS mode and a non-RLS mode concurrently, by different users.

Although data sets can be open in different modes at different times, all the data sets within a VSAM sphere must normally be opened in the same mode. A sphere is the collection of all the components—the base, index, any alternate indexes, and alternate index paths—associated with a given VSAM base data set. However, VSAM does permit mixed-mode operations on a sphere by different applications, subject to some CICS restrictions.

Effects

The tests and measurements described were carried out using RLS with key-sequenced data sets (KSDS). As described earlier in this topic, RLS is not suggested for use with entry-sequenced data sets (ESDS), as it can cause problems with performance and availability when you are adding records.

There is an increase in CPU costs when using RLS compared with function-shipping to a file-owning region (FOR) using MRO. When measuring CPU usage using the standard DSW workload, the following comparisons were seen:

- Switching from local file access to function-shipping across MRO cross-memory (XM) connections incurred an increase of 7.02 ms per transaction in a single CPC.
- Switching from MRO XM to RLS incurred an increase of 8.20 ms per transaction in a single CPC.
- Switching from XCF/MRO to RLS using two CPUs produced a *reduction* of 2.39 ms per transaction.
- Switching from RLS using one CPC to RLS using two CPUs there was no appreciable difference.

In terms of response times, the performance measurements showed that:

- Function-shipping with MRO XM is better than RLS, but this choice restricts function-shipping to within one MVS image, and prevents full exploitation of a Parallel Sysplex with multiple MVS images or multiple CPUs.
- RLS is better than function-shipping with XCF/MRO, when the FOR is running in a different MVS image from the AOR.

However, performance measurements on their own do not tell the whole story, and do not take account of other factors; for example:

- Because more applications need to share the same VSAM data, the load increases on the single FOR to a point where the FOR can become a throughput bottleneck. The FOR is restricted, because of the CICS internal architecture, to the use of a single TCB for user tasks, which means that a CICS region generally does not use multiple CPUs
- Session management becomes more difficult as more AORs connect to the FOR.

These negative aspects of using an FOR are resolved by using RLS, which provides the scalability lacking in a FOR.

Monitoring

Using RLS-access mode for VSAM files involves SMSVSAM as well as the CICS region issuing the file control requests. This choice means monitoring the performance of both CICS and SMSVSAM to get the full picture, using a combination of CICS performance monitoring data and SMF Type 42 records written by SMSVSAM:

CICS monitoring
For RLS access, CICS writes performance class records to SMF containing:
- RLS CPU time on the SMSVSAM SRB
- RLS wait time

SMSVSAM SMF data
SMSVSAM writes Type 42 records, subtypes 15, 16, 17, 18, and 19, providing information about coupling facility cache sets, structures, locking statistics, CPU usage, and so on. This information can be analyzed using RMF III post processing reports.

The following code is an example of the JCL that you can use to obtain a report of SMSVSAM data:

```
//RMFCF     JOB (accounting_information),MSGCLASS=A,MSGLEVEL=(1,1),CLASS=A
//STEP1     EXEC PGM=IFASMFDP
//DUMPIN    DD DSN=SYS1.MV2A.MANA,DISP=SHR
//DUMPOUT   DD DSN=&&SMF,UNIT=SYSDA,
//          DISP=(NEW,PASS),SPACE=(CYL,(10,10))
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
  INDD(DUMPIN,OPTIONS(DUMP))
  OUTDD(DUMPOUT,TYPE=000:255))
//POST      EXEC PGM=ERBRMFPP,REGION=0M
//MFPINPUT  DD DSN=&&SMF,DISP=(OLD,PASS)
//SYSUDUMP  DD SYSOUT=A
//SYSOUT    DD SYSOUT=A
//SYSPRINT  DD SYSOUT=A
//MFPMSGDS  DD SYSOUT=A
//SYSIN     DD *
 NOSUMMARY
 SYSRPTS(CF)
 SYSOUT(A)
 REPORTS(XCF)
/*
```

CICS file control statistics contain the typical information about the numbers of file control requests issued in the CICS region. They also identify which files are accessed in RLS mode, and provide counts of RLS timeouts and EXCP counts for RLS files. They do not contain any information about the SMSVSAM server, or its buffer usage, or its accesses to the coupling facility.

For more information about VSAM record-level sharing, see the following information:
- VSAM record-level sharing (RLS).
- IBM SupportPac CP13: IBM CICS TS record level sharing (RLS) performance study.
- IBM Redpaper™ publication IBM CICS Performance Series: CICS and VSAM RLS, REDP-4905.

Threadsafe file control applications

By default, CICS forces file control commands issued by threadsafe applications to run on the QR TCB. If you change the system initialization parameter **FCQRONLY** to specify NO, file control commands for local VSAM LSR or RLS files can run on an L8 or L9 TCB.

Using threadsafe file control can result in significant throughput improvements in CICS regions that have multiple processors available. Tasks currently running on an L8 or L9 TCB do not switch back to the QR TCB when the file control command is issued, but continue to run on the L8 or L9 TCB. These tasks benefit from greater concurrency and increased task throughput. Processor reduction and faster throughput is noticeable for threadsafe applications that combine file control commands with DB2 or WebSphere MQ requests.

To benefit from threadsafe file control, applications must meet the following requirements:
- The program resource must be defined with CONCURRENCY(THREADSAFE) or CONCURRENCY(REQUIRED).
- The file control commands that are issued must be to a local VSAM LSR or RLS file.
- The system initialization parameter **FCQRONLY=NO** must be specified for the CICS region where the file control commands run. **FCQRONLY=YES** is the default.

Threadsafe file control benefits CICS regions where the files are defined as local to the CICS region and are either VSAM LSR or RLS. From a file control perspective, in CICS regions with a mix of file types, consider specifying the system initialization parameter **FCQRONLY=NO**. Then define programs that access local VSAM LSR or RLS files with CONCURRENCY(THREADSAFE) and programs that access other file types with CONCURRENCY(QUASIRENT). If the files in a CICS region are not local VSAM LSR or RLS, use the default system initialization parameter **FCQRONLY=YES**.

Function shipped requests to file-owning regions (FORs)

If you function ship file control requests from application-owning regions (AORs) to file-owning regions (FORs), choose your setting for **FCQRONLY** as follows:
- For FORs at CICS TS 4.2 or later that use IP interconnectivity (IPIC) connections over TCP/IP, specify **FCQRONLY=NO** to optimize performance for those connections.
- For FORs that use MRO links or ISC over SNA connections, specify **FCQRONLY=YES** to optimize performance for those connections. Also use **FCQRONLY=YES** for all FORs earlier than CICS TS 4.2.

If an AOR function ships all its file control requests to FORs and has no local files, you can use the default **FCQRONLY=YES** for the AOR, because the region does not benefit from threadsafe file control. For AORs that have some local files, choose the setting for **FCQRONLY** depending on the file types in the region.

File control API costs

For read operations, the VSAM I/O cost is not included because the need to access DASD depends on the workload. For the read operation to complete, both the index and data must be accessed. If the index or data are not in a buffer, an I/O operation is required for each level of index and one for the data.

The relative number of instructions, in 1K instruction counts, for the I/O for each file type is as follows:
- 9.5 for a key-sequenced data set (KSDS)
- 9.5 for an entry-sequenced data set (ESDS)
- 8.2 for a relative record data set (RRDS)

READ

KSDS	ESDS	RRDS	Data Table (CMT)
3.0	2.4	2.2	First: 1.5 Subsequent: 1.1

READ UPDATE

Recoverable and nonrecoverable files are included in the READ UPDATE cost:

Table 16. Nonrecoverable files

KSDS	ESDS	RRDS
3.1	2.3	2.2

A recoverable READ UPDATE puts the before image into the log buffer which, if not subsequently written to primary storage, is written out before the REWRITE is completed.

KSDS	ESDS	RRDS
5.5	4.3	4.2

REWRITE

Recoverable and nonrecoverable files are included in the REWRITE cost. Every REWRITE has a data VSAM I/O associated with it.

Table 17. Nonrecoverable files

KSDS	ESDS	RRDS
10.2	10.1	10.1

A REWRITE of a recoverable file requires that the log buffer that containing the before image is written out. If the buffer has not already been written out since the READ UPDATE, the cost of writing the log buffer is incurred. When the before image has been hardened, the VSAM I/O takes place. At the end of the transaction, there are additional costs involved in sync pointing if recoverable resources were updated. See “Sync pointing” on page 223.

KSDS	ESDS	RRDS
10.4	10.3	10.3

WRITE

The cost for WRITE includes nonrecoverable files and recoverable files. Every WRITE has a data VSAM I/O associated with it. The index needs to be written only when a control area split occurs.

Table 18. Nonrecoverable files

KSDS	ESDS	RRDS
12.9	11.1	10.9

Every WRITE has a hidden READ associated with it to ensure that the record is not already present in the file. This under-the-cover READ could incur the cost of I/Os if the index, the data, or both are not in the buffer. Each WRITE to a recoverable file requires that the log buffer containing the data image has been written out before the VSAM I/O takes place.

At the end of the transaction, there are additional costs involved in sync pointing if recoverable resources were updated. See "Sync pointing" on page 223.

Table 19. Recoverable files

KSDS	ESDS	RRDS
14.9	13.1	12.9

DELETE

You cannot delete from an ESDS record file.

Table 20. Nonrecoverable files

KSDS	RRDS
12.5	11.5

At the end of the transaction, additional costs are involved in sync pointing if recoverable resources were updated. See "Sync pointing" on page 223.

Table 21. Recoverable files

KSDS	RRDS
14.5	13.5

Browsing

STARTBR	READNEXT	READPREV	RESETBR	ENDBR
3.1	1.5	1.6	2.6	1.4

UNLOCK

The path length for **EXEC CICS UNLOCK** is 0.7.

Chapter 15. Database management for performance

You can tune a number of aspects of database management in order to improve performance.

Setting DBCTL parameters

A number of parameters are required to assist with DBCTL performance. These include **MINTHRD** and **MAXTHRD**, which are specified in the DRA startup table (DFSPZP) and DEDB parameters (**CNBA**, **FPBUF**, and **FPBOF**), which are defined during DBCTL system generation or at DBCTL initialization.

For more information about the DBCTL parameters and tuning a CICS-DBCTL system, see Specifying numbers of threads in Improving performance and DEDB performance and tuning considerations in Improving performance.

Tuning the CICS DB2 attachment facility

The CICS DB2 attachment facility provides a multithread connection to DB2. The DB2CONN, DB2ENTRY, and DB2TRAN definitions of the CICS DB2 attachment facility define the authorization and access attributes on a transaction and transaction group basis. You can optimize performance between CICS and DB2 by adjusting the transaction class limits, MXT system parameters of CICS, and the THREADWAIT, TCBLIMIT, THREADLIMIT, and PRIORITY attributes of DB2CONN and DB2ENTRY.

A number of topics provide more information about the CICS DB2 attachment and performance considerations:
- Defining the CICS DB2 connection explains the recommendations for defining the CICS DB2 connection for optimum performance.
- How threads are created, used, and terminated in Configuring explains threads and the use of the THREADWAIT, TCBLIMIT, and THREADLIMIT parameters with DB2.
- Application design and development considerations for CICS DB2 has recommendations for application design.
- Tuning a CICS application that accesses DB2 has recommendations for tuning CICS DB2 applications.

In summary, the objectives in tuning the CICS attachment facility are to:
- Optimize the number of threads in the connection.

  The total number of threads in the connection, and the number of threads for each dedicated entry and the pool must be optimized. A larger number of threads than is needed requires additional processor time to dispatch the TCBs and additional storage for plans, data, and control blocks. If an insufficient number of threads is defined, response time increases.
- Optimize the assignment and reuse of threads.

  Reusing threads avoids the thread creation and termination process, including plan allocation and authorization checks. Thread creation and termination represent a significant part of the processing time for a simple transaction. Thread reuse can be measured using CICS DB2 statistics.

Limit conversational transactions either through transaction classes or by using a dedicated DB2ENTRY (THREADLIMIT greater than 0) with THREADWAIT=YES specified. Otherwise, they tie up the pool. Do not allow conversational transactions to use the pool.

- For pool and entry threads, choose the priority assigned to the subtask thread TCBs, using the PRIORITY parameter.

  The **PRIORITY** parameter controls the priority of the CICS open L8 thread TCBs relative to the CICS main TCB (QR TCB). There are three options: PRIORITY=HIGH, PRIORITY=LOW, and PRIORITY=EQUAL. See RDO resources for more information.

  When PRIORITY=HIGH is specified, transactions run at a higher priority than CICS, saving virtual storage, releasing locks, and avoiding other transactions deadlocking or timing out. However, if all threads are specified with PRIORITY=HIGH, CICS itself might be at too low a priority, so for example, a complex SQL call could spend a long time in DB2, and the CICS TCB might not be dispatched.

  Set PRIORITY=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 PRIORITY=LOW or EQUAL for other transactions. If the CPU usage per call is high, you should not set PRIORITY=HIGH.
- Choose the best authorization strategy to avoid or minimize the process of signon by each thread.
- Minimize the number of DB2ENTRYs. Use wildcarding and dynamic plan selection where relevant to combine appropriate transactions in an entry. Allow low use transactions to default to the pool. However, it should be noted that defining transaction IDs using wildcard characters removes the ability to collect CICS DB2 statistics on a per transaction basis as statistics are collected for each DB2ENTRY which will now represent a group of transactions.

For information about tuning DB2 tables and the DB2 subsystem, and for general considerations when tuning a DB2 application, see the *DB2 for z/OS: Programming for Java*.

Selecting authorization IDs for performance and maintenance

A process that connects to or signs on to DB2 must provide one or more DB2 short identifiers, called authorization IDs, that can be used for security checking in the DB2 address space. Every process must provide a primary authorization ID, and it can optionally provide one or more secondary authorization IDs. CICS transactions that acquire a thread into DB2 are considered as processes, and must provide authorization IDs.

Providing authorization IDs to DB2 for the CICS region and for CICS transactionsIn the *CICS DB2 Guide* tells you how to choose and set up the authorization IDs that a CICS transaction passes to DB2 when the thread used by the transaction signs on to DB2. The authorization IDs for a transaction are determined by attributes in the resource definition for the thread that the transaction uses. For entry threads, this is the DB2ENTRY definition, and for pool threads or command threads, this is the DB2CONN definition.

When choosing the type of authorization ID that a CICS transaction will use, you should take into account performance and maintenance considerations.

Performance considerations for authorization IDs

From the point of view of performance, choosing one of the options USERID, OPID, TERM, TX or GROUP on the AUTHTYPE attribute means that any CICS transaction using a DB2 thread is likely to have a different authorization ID from the last transaction that used the thread. This causes sign-on processing to occur. Choosing the SIGN option, or using the AUTHID attribute instead of the AUTHTYPE attribute, means that CICS transactions will have the same authorization ID. If the transactions using a thread have the same authorization ID, sign-on processing can be bypassed.

However, although the options USERID, OPID, TERM, TX or GROUP have disadvantages for performance, they make DB2's security checking more granular. For example, if a transaction's thread is defined with AUTHTYPE(USERID), DB2's security checking uses the CICS user ID of the individual that is using the transaction. If a transaction's thread is defined with AUTHTYPE(SIGN), DB2's security checking uses the SIGNID that has been defined for the whole CICS region, so DB2 is only checking that the CICS region is permitted to access DB2 resources. If you do use one of the options that gives the same authorization ID for all transactions, you should use CICS transaction-attach security to restrict access to transactions (see Controlling users' access to DB2-related CICS transactions in the *CICS DB2 Guide*).

An alternative solution for plans is to use a GRANT command in DB2 to give EXECUTE authority on a plan to PUBLIC, because this also causes sign-on processing to be bypassed. DB2 ignores the changed authorization ID. This is not quite as efficient as using a constant authorization ID and transaction id, because some processing still takes place in the CICS DB2 attachment facility. Security considerations for your DB2 subsystem could prevent the use of this solution, as it allows no security checking for the plan within DB2.

Maintenance considerations for authorization IDs

From the point of view of maintenance, when you use the options USERID, OPID, TERM, TX or GROUP for authorization IDs, you need to grant permissions in DB2 to a greater number of authorization IDs. For example, if a CICS transaction executes a plan in DB2, and the transaction's thread is defined with AUTHTYPE(USERID), you need to grant permission to use the plan in DB2 to all the CICS user IDs of individuals who can use the transaction. If you use the SIGN option, or use the AUTHID attribute instead of the AUTHTYPE attribute, you need to grant permissions to fewer authorization IDs.

However, as already mentioned, using a limited range of authorization IDs makes DB2's own security checking less granular. If your priority is security, but you are concerned about high levels of maintenance in your DB2 system, a possible solution is to set up secondary authorization IDs for CICS users. Providing secondary authorization IDs for CICS transactions in the *CICS DB2 Guide* tells you how to do this. You can create a RACF group, and connect your CICS users to this RACF group. Use the GROUP attribute of the DB2ENTRY definition for the thread used by the transaction, so that the RACF group is one of the secondary IDs that is passed to DB2. Then grant DB2 permissions to the RACF group. To remove a CICS user's DB2 permissions, disconnect them from the RACF group. If you use this solution, DB2's security checking can ensure that individual CICS users are authorized to access resources within DB2, but you do not have to specifically grant permission to each CICS user ID.

Logging

Because logging costs contain some of the variable costs incurred by synchronous accesses to the coupling facility, they are documented here in terms of milliseconds of CPU time.

The measurements have been taken on a 9672-R61 with a 9674-R61 coupling facility; they can be scaled to any target system, using the IT Relative Ratios (ITRRs) published in the *IBM Large System Performance Report*. This can be accessed through the IBM System/390® web page (http://www.s390.ibm.com), more specifically, at http://www.s390.ibm.com/lspr/lspr.html.

When looking at the cost of accessing recoverable resources, the cost of writing the log buffer to primary storage has been separated from the API cost. FORCE and NOFORCE are the two types of write operations to the system log buffer.

- The FORCE operation requests that the log buffer is written out and is made non-volatile. The transaction that made this request is suspended until the process completes. The log is not written out immediately but is deferred using an internal algorithm. The first forced write to the log sets the clock ticking for the deferred log flush. Subsequent transactions requesting log forces will put their data in the buffer and suspend until the original deferred time has expired. This permits buffering of log requests and it means that the cost of writing the log buffer is shared between many transactions.
- The NOFORCE operation puts the data into the log buffer, which is written to primary storage when a FORCE operation is requested or the buffer becomes full.

The cost of writing a log buffer varies, depending on which of the following situations applies:

- The write is synchronous to the coupling facility
- The write is asynchronous to the coupling facility
- A staging data set is being used
- DASD-only logging is being used

Synchronous writes to the coupling facility
Writes of less than 4 K in size are generally synchronous. A synchronous write uses a special instruction that accesses the coupling facility directly. The instruction lasts for as long as it takes to access the coupling facility and return. This access time, known as the *CF Service Time*, depends on both the speed of the coupling facility and the speed of the link to it. CF Service Times can be monitored using RMF III, as shown in Figure 21 on page 231. For synchronous writes, the CPU cost of the access changes as the CF Service Time changes; this is not true of asynchronous writes.

Asynchronous writes to the CF
Asynchronous writes do not use the same instruction used by synchronous writes. A CICS task that does an asynchronous log write gives up control to another task, and the operation is completed by the logger address space.

For more information about logging, see Chapter 16, “CICS logging and journaling: Performance and tuning,” on page 225.

Sync pointing

The sync point cost needs to be factored into the overall transaction cost. The amount of work at sync point varies according to the number of different types of resource managers involved during the unit of work (UOW). Therefore, the cost can vary.

Typically, a sync point calls all the resource managers that have been involved during the UOW. These might have to place data in the log buffer before it is written out. For example, recoverable transient data (TD) defers putting data into the log buffer until a sync point. Recovery manager itself puts commit records into the log buffer and requests a forced write. For these reasons it is difficult to give a precise cost for a sync point, but the following information should be used as a guide:

A sync point can be split as follows:

Part	Value
Basic cost	5.0
Put commit records in the log buffer	2.0
For each RM used in UOW	2.5
Write log buffer	See “Logging” on page 222

This table shows sync point costs, in 1K instruction units, for local resources only. If distributed resources are updated, communication costs must added.

If no recoverable resources have been updated, the only cost is the transaction termination cost:

Transaction cost	Assembler	COBOL
Termination	6.2	10.0

Note: The transaction initialization cost is calculated from the start of transaction attach to the start of the CICS application code. If recoverable resources have been updated, the sync pointing cost must be added to the termination cost.

Chapter 16. CICS logging and journaling: Performance and tuning

Individual CICS log streams can use either coupling facility log structures or the CICS log-manager-supported DASD-only option of the MVS system logger. You can tune the performance of the log manager in a number of ways.

For more information about the types of storage used by CICS log streams, see Defining the logger environment for CICS journaling in the *CICS Transaction Server for z/OS Installation Guide*.

For information about how you can define each log stream (based on its usage) when you use coupling facility log structures, see Coupling facility or DASD-only? in the *CICS Transaction Server for z/OS Installation Guide*. For information about the relative performance of coupling facility and DASD-only log streams, see “Logging” on page 222.

If you use a coupling facility, you can use a standalone model. Alternatively, you can use the integrated coupling migration facility (ICMF) to provide the services of a coupling facility in a logical partition (LPAR). This means that the coupling facility and MVS are not failure-independent, thereby requiring the use of staging data sets.

For additional advice and examples relating to performance and tuning for logging, see the following documents and subtopics:

- The IBM Redbooks publication Systems Programmer's Guide to: z/OS System Logger, SG24-6898. This document provides a thorough explanation of the z/OS System Logger, and explains how it should be set up for optimum performance with CICS and other exploiters.
- The IBM Redpaper Performance Considerations and Measurements for CICS and System Logger, REDP-3768. This document, which was written in support of the Redbook publications, supplies additional guidance on the interactions between CICS and z/OS System Logger, provides examples of different CICS and System Logger configurations, and demonstrates the tuning process.
- The IBM support document Useful CICS Logger information. This document provides links to two presentations dealing with performance evaluation and troubleshooting for CICS and z/OS System Logger.
- Defining a couple data set for system logger in z/OS Management Facility Configuration Guide.
- Examples of using the IXCMIAPU utility in z/OS MVS Setting Up a Sysplex.

The CICS log manager

The CICS log manager provides facilities for the creation, control, and retrieval of journals when CICS is running. Journals are intended to record, in chronological order, any information that you might later need to reconstruct data or events. For example, you can create journals to act as audit trails; to record database updates, additions, and deletions for backup purposes; or to track transaction activity in the system.

The CICS log manager controls all logging and journaling using services provided by the MVS system logger. The CICS log manager supports:
- The CICS system log
- Forward recovery logs
- Auto-journals for file control and terminal control operations
- User journals

The MVS system logger provides:
- Media management and archiving
- Log data availability through direct, and sequential, access to log records.

Log stream storage

A log stream is a sequence of data blocks, with each log stream identified by its own log stream identifier—the log stream name (LSN). The CICS system log, forward recovery logs, and user journals map onto specific MVS log streams. CICS forward recovery logs and user journals are referred to as general logs, to distinguish them from system logs.

Each log stream is a sequence of blocks of data, which the CICS log manager internally partitions over three different types of storage:
1. Primary storage, which holds the most recent records written to the log stream. Primary storage can consist of either:
   - A structure within a coupling facility. (The use of a coupling facility allows CICS regions in different MVS images to share the same general log streams.) Log data written to the coupling facility is also copied to either a data space or a staging data set.
   - A data space in the same MVS image as the system logger. Log data written to the data space is also copied to a staging data set.
2. Auxiliary storage—when the primary storage for a log stream becomes full, the older records automatically spill into auxiliary storage, which consists of data sets managed by the storage management subsystem (SMS). Each log stream, identified by its log stream name (LSN), is written to its own log data sets.
3. Tertiary storage—a form of archive storage, used as specified in your hierarchical storage manager (HSM) policy. Optionally, older records can be migrated to tertiary storage, which can be either DASD data sets or tape volumes.

Figure 19 on page 227 and Figure 20 on page 228 show the types of storage used by the CICS system logger.

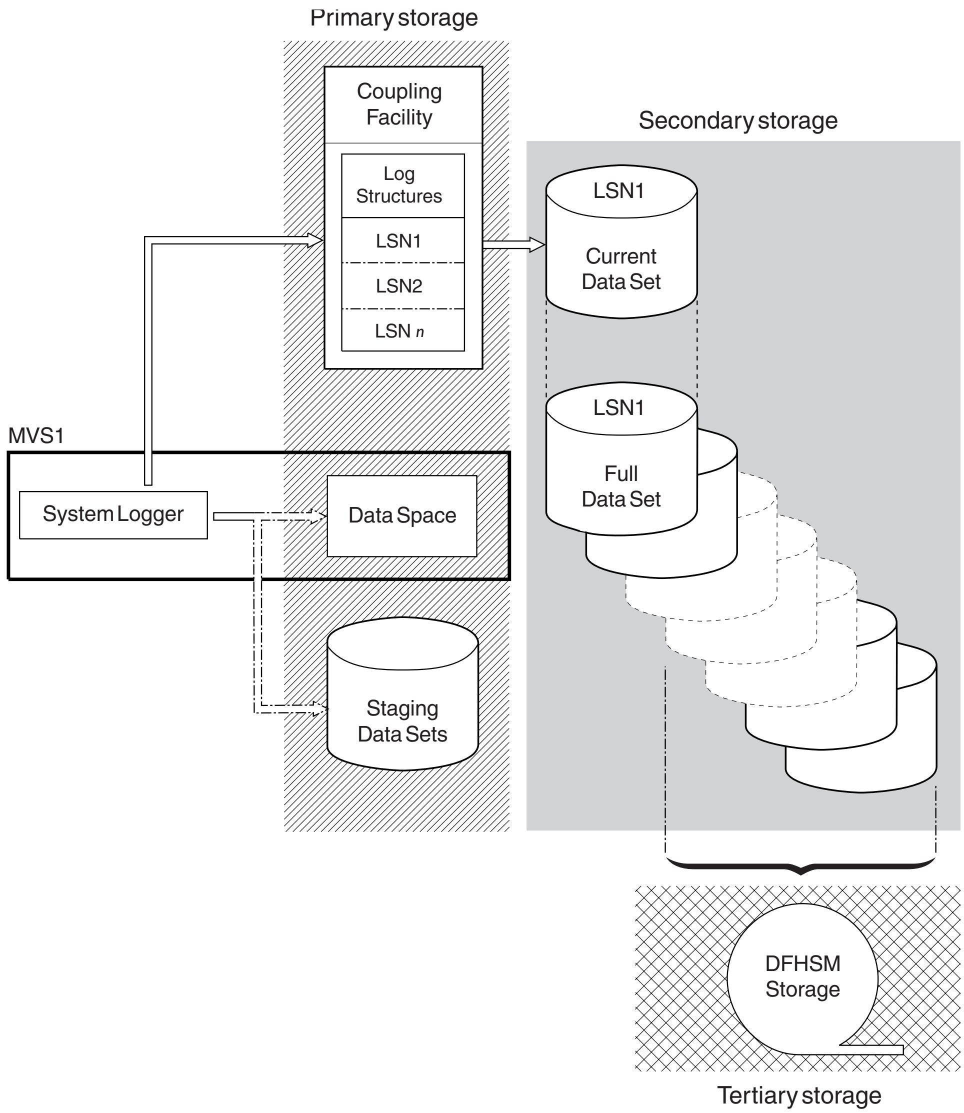


Figure 19. The types of storage used by the MVS system logger. This diagram shows a log stream that uses a coupling facility. Primary storage consists of space in a structure within the CF, and either staging data sets or a data space in the same MVS image as the system logger.

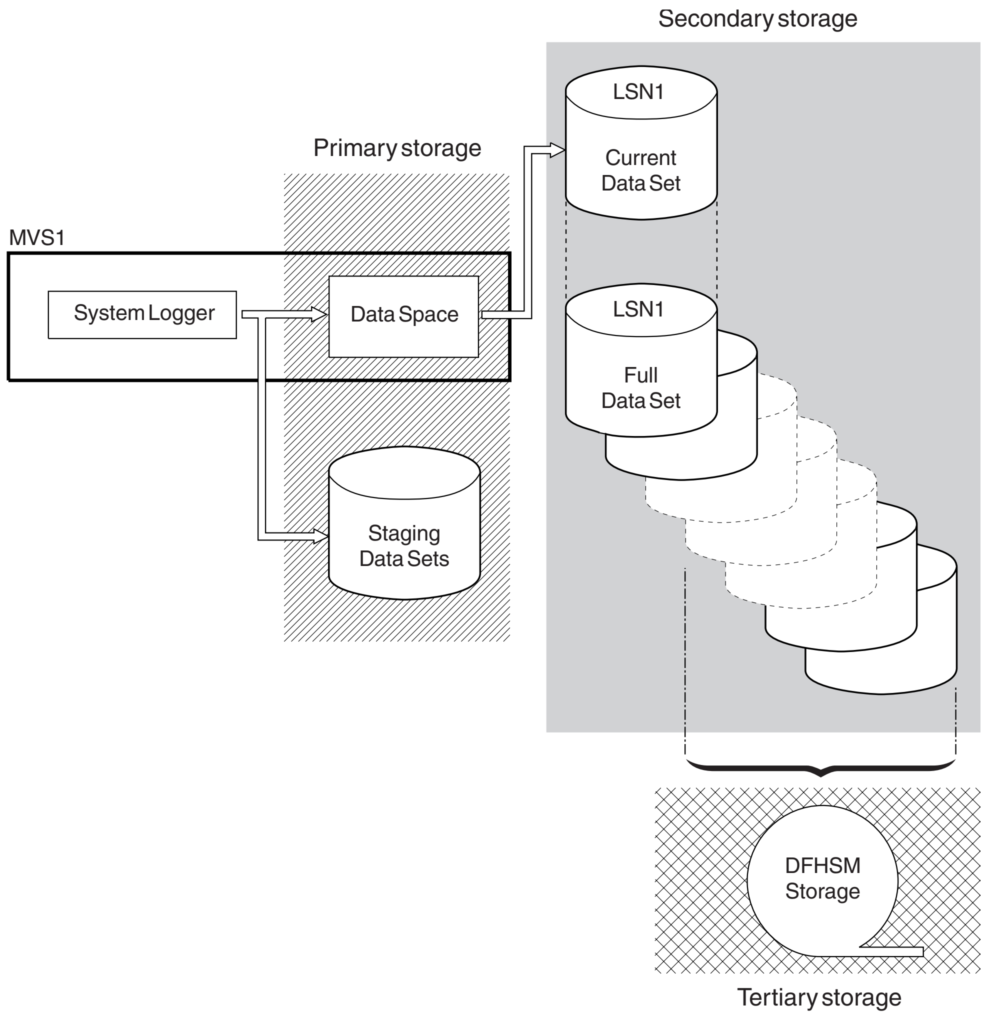


Figure 20. The types of storage used by the MVS system logger. This diagram shows a log stream that uses DASD-only logging. Primary storage consists of a data space in the same MVS image as the system logger, and a single staging data set.

Journal records

Journal records are written to a log stream either directly from a user application program or from a CICS management program on behalf of a user application.

Journal records can be written from a user application using the WRITE JOURNALNAME API command. You enable or disable a journal from an application program with the SET JOURNALNAME API command.

Access to journaled data in log streams is provided through an MVS subsystem interface (SSI), LOGR. Your existing user programs can read the general log streams, providing you specify the **SUBSYS** parameter and supporting options on the DD for log streams in your batch job JCL. If you specify the LOGR subsystem name on the **SUBSYS** parameter, LOGR can intercept data set open and read requests at the SSI and convert them into log stream accesses.

Depending on the options specified on the **SUBSYS** parameter, general log stream journal records are presented in one of two ways:

- In the record format used at CICS/ESA 4.1 and earlier, for compatibility with older utilities (selected by the COMPAT41 option)
- In the CICS Transaction Server for z/OS format for newer or upgraded utilities that needed to access log record information.

CICS system log records are available only in the CICS Transaction Server for z/OS format, so you must ensure that any utilities that handled system log records in releases before CICS Transaction Server for z/OS are converted to handle this format.

Journal records can be read offline by user-written programs. You can generate the DSECTs that such programs require by including certain statements in the program code:

- For records in the CICS Transaction Server for z/OS format on general logs, offline user-written programs can map journal records by including an INCLUDE DFHLGGFD statement. This statement generates the assembler version of the DSECT.
- For records formatted with the COMPAT41 option, offline user-written programs can map journal records by issuing the DFHJCR CICSYST=YES statement, which results in the DFHJCRDS DSECT being included in the program.

  The generated DSECT is the same as the DSECT that is obtained for CICS programs by the COPY DFHJCRDS statement. The only difference is that the fields are not preceded by a CICS storage accounting area. The DSECT is intended to map journal records directly in the block, rather than in a CICS storage area.

Monitoring the logger environment

CICS collects statistics on the data written to each journal and log stream; this data can be used to analyze the activity of a single region. However, because general log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

About this task

The MVS system logger writes SMF Type 88 records containing statistics for each connected log stream. MVS supplies in SYS1.SAMPLIB a sample reporting program, IXGRPT1, that you can use as supplied, or modify to meet your requirements. Alternatively, you can use some other SMF reporting program. For information about the SMF Type 88 records and the sample reporting program, see *z/OS MVS System Management Facilities (SMF)*.

The main events to monitor routinely are as follows:

- For coupling facility log streams, the number of “structure full” events
- For DASD-only log streams, the number of “staging data set full” events.

If these events occur frequently, this indicates that the logger cannot write data to auxiliary storage quickly enough to keep up with incoming data, which causes CICS to wait before it can write more data.

Procedure

1. Consider the following solutions to resolve problems that occur as a result of event-full conditions:

a. Increase the size of primary storage (that is, the size of the coupling facility structure or, for a DASD-only log stream, the size of the staging data set), in order to smooth out spikes in logger load.
b. Reduce the data written to the log stream by not merging so many journals or forward recovery logs on to the same stream.
c. Reduce the **HIGHOFFLOAD** threshold percentage, the point at which the system logger begins offloading data from primary storage to offload data sets.
d. Review the size of the offload data sets. Offload data sets must be large enough to avoid too many “DASD shifts”—that is, new data set allocations. Aim for no more than one DASD shift per hour. You can monitor the number of DASD shifts using the SMF88EDS record.
e. Examine device I/O statistics for possible contention on the I/O subsystem used for offload data sets.
f. Use faster DASD devices.

The best CICS system logs performance is achieved when CICS can delete log tail data that is no longer needed before it is written to auxiliary storage by the MVS system logger. To monitor that this is being achieved, your reporting program can examine the values in the SMF88SIB and SMF88SAB SMF Type 88 records, which provide helpful information relating to log data.

SMF88SIB
Data deleted from primary storage without first being written to DASD offload data sets. For a system log stream, this value is normally high in relation to the value of SMF88SAB. For a general log stream, this value is normally zero.

SMF88SAB
Data deleted from primary storage after being written to DASD offload data sets. For a system log stream, this value is normally low in relation to the value of SMF88SIB. For a general log stream, this value is normally high.

Note: In any SMF interval, the total number of bytes deleted from primary storage (SMF88SIB plus SMF88SAB) might not match the total number of bytes written to auxiliary storage. Data is only written to offload data sets and then deleted from primary storage when the **HIGHOFFLOAD** threshold limit is reached.

2. If the SMF88SAB record frequently contains high values for a CICS system log:
   a. Check that RETPD=*dddd* is not specified on the MVS definition of the log stream. For information about the MVS **RETPD** parameter, see Managing auxiliary storagein the *CICS Transaction Server for z/OS Installation Guide*.
   b. Check that no long-running transactions are making recoverable updates without syncpointing.
   c. Consider increasing the size of primary storage.
   d. Consider increasing the **HIGHOFFLOAD** threshold value.
   e. Consider reducing the value of the **AKPFREQ** system initialization parameter.

Writing data to the coupling facility: Performance considerations

At the application design level you must consider that the average block size written to the coupling facility affects the performance of the CICS log manager.

When the average block size of data being written to the coupling facility is less than 4 KB, the write request is processed synchronously. The operation is synchronous to CICS, as is the instruction used to access the coupling facility, in that it runs for as long as it takes to place the data in the structure. For this reason, it is unwise to mix fast processors with slow coupling facilities. If the access time to a particular coupling facility remains constant, then for synchronous accesses, the faster the processor the more processor cycles are used by the request.

When the average block size of data being written to the coupling facility is greater than 4 KB, the write request is processed asynchronously; the CICS task gives up control and the MVS system logger posts the event control block (ECB) when the write request has been satisfied. This can result in an asynchronous request taking longer to complete than a synchronous one.

Synchronous requests might be changed by the subsystem into asynchronous requests if necessary—for example, if the subchannel is busy. Changed requests show on an RMF III report as CHNGD. Figure 21 shows an extract from an RMF report showing the numbers of synchronous and asynchronous writes to a coupling facility structure. The report gives the system name, the total number of requests, and the average number of requests per second. For each type of request, it gives the number of requests, the percentage of all requests that this number represents, the average service time, and the standard deviation.

```
STRUCTURE NAME = LOG_FV_001        TYPE = LIST
          # REQ    -------------- REQUESTS -------------
SYSTEM    TOTAL             #     % OF  -SERV TIME(MIC)-
NAME      AVG/SEC           REQ    ALL     AVG   STD_DEV

MV2A      15549    SYNC     15K  95.3%   476.1    339.6
          27.87    ASYNC   721    4.6%  3839.0   1307.3
                   CHNGD    12    0.1%  INCLUDED IN ASYNC
```

Figure 21. RMF report showing numbers of synchronous and asynchronous writes to a coupling facility

Note: This applies only to log streams that use coupling facility structures.

Defining the number of log streams: Performance considerations

Coupling facility space is divided into structures by the coupling facility resource management (CFRM) policy; the maximum is 255 structures. Multiple log streams can use the same structure. Ensure that log streams used by applications that write similar sized data records share the same structure. The reasons for this relate to the values defined in the **AVGBUFSIZE** and **MAXBUFSIZE** parameters on the structure definition.

Generally, the more log streams per structure, the more difficult it is to tune the various parameters that affect the efficiency and performance of the CICS log manager.

When a coupling facility structure is defined, it is divided into two areas: one holds list entries, and the other holds list elements.

List elements are units of logged data and are either 256-bytes or 512-bytes long. List entries are index pointers to the list elements. There is one list entry per log record. There is at least one element per log record.

If you define **MAXBUFSIZE** with a value greater than 65276, data is written in 512-byte elements. If you define **MAXBUFSIZE** with a value less than, or equal to, 65276, data is written in 256-byte elements. The maximum value for this parameter is 65532.

The proportion of the areas occupied by the list entries and the list elements is determined by a ratio calculated as follows:

```
AVGBUFSIZE / element size
```

The resulting ratio represents the ratio, *nn*: 1, where *nn* represents element storage, and 1 represents entry storage. This is subject to a minimum of 1:1.

This ratio has performance significance because it can be inappropriate for a combination of many different applications with different logging requirements and behavior.

Element/entry ratio and the number of log streams per structure

AVGBUFSIZE is set at the structure level and dictates the ratio for the whole structure. If many applications write significantly differing amounts of data to their log streams at significantly differing intervals, some applications might experience unexpected DASD offloading, incurring increased processor usage.

The DASD offloading is unexpected because the log stream might not yet have reached the **HIGHOFFLOAD** threshold. Generally, the greater the number of log streams per structure, the greater the chance that the element/entry ratio is inappropriate for certain applications that use the log streams.

Each log record places an entry in the list entry area of the structure, and the data is loaded as one or more elements in the list element area. If the list entry area exceeds 90% of its capacity, all log streams are offloaded to DASD. DASD offloading commences regardless of the current utilization of the log stream, and continues until an amount of data equal to the difference between the **HIGHOFFLOAD** threshold and the **LOWOFFLOAD** threshold has been offloaded.

For example, the list entry area might exceed 90% of its capacity while log stream A is only 50% used. The **HIGHOFFLOAD** threshold is 80% and the **LOWOFFLOAD** threshold is 60%. Even though log stream A has not reached its **HIGHOFFLOAD** threshold, or even the **LOWOFFLOAD** threshold, data is offloaded until 20% of the log stream has been offloaded. This is the difference between 80% and 60%. After the offloading operation has completed, log stream A is at 30% utilization (50% minus 20%).

Thus, the log stream used by an application that issues few journal write requests might be offloaded to DASD because of frequent journal write requests by other applications that are using other log streams in the same structure.

However, if multiple log streams share the same structure, a situation where list entry storage reaches 90% utilization occurs only where all the log streams have a similar amount of logging activity.

Dynamic repartitioning and the frequency of DASD offloading

The space in a coupling facility structure is dynamically partitioned between all the log streams connected to the structure. As more log streams connect, DASD offloading might occur more often.

Whenever a log stream connects to, or disconnects from, a coupling facility structure, the structure undergoes dynamic repartitioning. This means that the space in the structure is partitioned between all the log streams connected to the structure. As more log streams connect, less space is allocated to each log stream. The result can be a higher frequency of DASD offloading, because reduced log stream space means that the log stream **HIGHOFFLOAD** threshold percentages are reached more often.

A value of 64000 for **MAXBUFSIZE** is appropriate for most environments.

If **MAXBUFSIZE** is set to greater than 65276, the element size is 512 bytes. With a 512-byte element, space might be unused and therefore wasted because of padding to the end of the last element for the log record. This situation is less likely when records are larger and systems are busier.

AVGBUFSIZE and **MAXBUFSIZE** are parameters for use in the IXCMIAPU program, which you run to define coupling facility structures. For more information, see Administrative data utility in z/OS MVS Setting Up a Sysplex.

The following facilities are available to monitor the data traffic to log streams on structures, and from log streams to DASD:
- The CICS log stream statistics. These provide a range of statistical information including a value for average bytes written per write, which you can calculate by dividing the total bytes value by the total writes value. This can help you to tune the value for **AVGBUFSIZE**.
- Statistics provided by RMF, including a value 'elements per entry', which you can calculate by dividing the total number of elements value by the total number of entries value. You can check the activity in element units on the log stream. RMF also informs you of the proportion of requests, per structure, that have been processed synchronously and asynchronously. You can isolate structures that hold synchronously processed log stream requests from those that hold asynchronously processed log stream requests.
- SMF88 records. These provide a range of statistical information, including the number of bytes offloaded.

LOWOFFLOAD and HIGHOFFLOAD parameters on log stream definition

Data from a log stream can be offloaded to DASD data sets when the log stream use (in the coupling facility or the staging data set) reaches its **HIGHOFFLOAD** limit. The amount of data offloaded is determined by using the **LOWOFFLOAD** limit.

The **HIGHOFFLOAD** limit is specified when the log stream is defined.

This information is relevant if you are using log streams that use coupling facility structures. However, much of the guidance also applies to DASD-only log streams.

For more information about DASD-only log streams, see “DASD-only logging” on page 238.

For a system log, all records that have been marked for deletion are physically deleted; if, after this has been done, the **LOWOFFLOAD** limit has not been reached, the oldest active records are offloaded to DASD until **LOWOFFLOAD** is reached. For a general log, the oldest data is offloaded to DASD until the **LOWOFFLOAD** limit is reached.

There are also situations where offloading of data from the log stream data set occurs although the **HIGHOFFLOAD** threshold (and **LOWOFFLOAD** threshold in some circumstances) of the log stream has not been reached:

- When the **HIGHOFFLOAD** threshold is reached in the staging data set. If the size of the staging data set is proportionally smaller than the log stream, the **HIGHOFFLOAD** threshold is reached on the staging data set before it is reached on the log stream data set.
- When the list entry area of the log stream reaches 90% of its capacity.

In these situations, the amount of data offloaded from the log stream is determined as follows:

```
(Current utilization or HIGHOFFLOAD, whichever is the greater) - LOWOFFLOAD
```

This is the percentage of the log stream data set that is offloaded.

HIGHOFFLOAD and **LOWOFFLOAD** are parameters for use in the IXCMIAPU program that you run to define log stream models and explicitly named individual log streams. For more information, see Administrative data utility in z/OS MVS Setting Up a Sysplex.

SMF88 records and RMF provide a range of statistical information that helps you in the tuning of these parameters.

The primary system log

When an activity keypoint happens, CICS deletes the tail of the primary system log, DFHLOG. This means that data for completed units of work older than the previous activity keypoint is deleted. Data for each incomplete unit of work older than the previous activity keypoint is moved onto the secondary system log, DFHSHUNT, provided that the UOW has done no logging in the current activity keypoint interval.

To minimize the frequency of DASD offloading, try to ensure that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is always in the coupling facility structure. To avoid offloading this data to DASD, you can use these settings:

- Set **HIGHOFFLOAD** to 80.
- Minimize the amount of log data produced between activity keypoints by specifying a low value on the **AKPFREQ** parameter, for example, a value of 4000.
- Ensure that the value of **LOWOFFLOAD** is greater than the space required for the sum of:
  1. The system log data generated during one complete activity keypoint interval
  2. The system log data generated (between sync points) by your longest-running transaction.

  Use one of the following formulas to calculate a value for **LOWOFFLOAD**:

  ```
  LOWOFFLOAD = ((trandur * 90) / (akpintvl + trandur)) + 10
  [where RETPD=0 is specified]
  ```

  or

  ```
  LOWOFFLOAD = (trandur * 90) / (akpintvl + trandur)
  [where RETPD=dddd is specified]
  ```

  where:

– akpintvl is the interval between activity keypoints. It varies according to workload and its calculation is based on peak workload activity, as follows:

```
akpintvl = AKPFREQ / ((N1 * R1) + (N2 * R2*) + (Nn * Rn))
```

where:

- N1, N2 ... Nn is the transaction rate for each transaction (trans/sec)
- R1, R2 ... Rn is the number of log records written by each transaction

– trandur is the execution time (between sync points) of the longest-running transaction that runs as part of the normal workload.

If this duration is longer than the akpintvl value, you can either:

- Increase the value of **AKPFREQ**, thus increasing the value of akpintvl (providing this does not result in an unacceptably large coupling facility structure size).
- Change the application logic to cause more frequent sync points.
- Calculate a structure size based on a shorter transaction duration, and accept that DASD offloading occurs when the long-running transaction is used.

A good empirical range for the DFHLOG **LOWOFFLOAD** parameter value is between 40% and 60%. A value that is too low can result in physical offloading of log data from primary to auxiliary storage after the MVS Logger offload process has completed physical deletion of any unwanted log data during offload processing. Conversely, too high a value might mean that subsequent offload processing occurs more frequently, as less space is freed up from primary storage during an offload operation.

If the results of the calculation from the formula do not lie within the range of 40% to 60%, it might be that your workload has unusual values for trandur or akpintvl.

Review log stream definition values (such as **LOWOFFLOAD**) after analysis of information such as statistics from MVS logger SMF 88 records.

General logs

The recommendations for forward recovery logs and user journals are different to those for the system log. There is no requirement here to retain logged data in the coupling facility structure. Rather, due to the typical use of such data, you might only need a small structure and offload the data rapidly to DASD. If so, default **HIGHOFFLOAD** to 80 and **LOWOFFLOAD** to 0.

Tuning the size of staging data sets

MVS keeps a second copy of data written to the coupling facility in a data space, for use when rebuilding a coupling facility in the event of an error. This is satisfactory as long as the coupling facility is failure-independent (in a separate CPC and non-volatile) from MVS.

Where the coupling facility is in the same CPC, or uses volatile storage, the MVS system logger supports staging data sets for copies of log stream data that would otherwise be vulnerable to failures that impact both the coupling facility and the MVS images.

Elements (groups of log records) are written to staging data sets in blocks of 4 KB (not in 256-byte or 512-byte units as for log stream data sets).

Use the following formulas to help you tune the size of your staging data sets:

```
staging data set size= (NR * AVGBUFSIZE rounded up to next unit of 4096)
```

where NR is the number of records to fill the coupling facility structure. This can be calculated as follows:

```
NR = coupling facility structure size / (AVGBUFSIZE rounded up to next element)
```

Ensure that the coupling facility structure and staging data set can hold the same number of records. Staging data sets are subject to the same offloading thresholds as log streams are. It is sensible, therefore, to ensure as far as possible that offloading activity will be at the same frequency.

It is generally better to overestimate, rather than underestimate, staging data set size. To calculate staging data set size to accommodate the maximum number of records (where there is one record per element), use the following formulas:

Where element size is 512-bytes:

```
maximum staging data set size = 8 * coupling facility structure size
```

Where element size is 256-bytes:

```
maximum staging data set size = 16 * coupling facility structure size
```

Investigate using DASD FastWrite facilities with a view to storing data in the DASD cache, as opposed to writing it directly to the staging data set. This also enables a faster retrieval of data should it be required. Be aware, however, that if you fill the cache, data is also then written out to the staging data set whenever data is written to the cache.

The activity keypoint frequency (AKPFREQ)

The activity keypoint frequency value, AKPFREQ, specifies the number of write requests to the CICS system log stream output buffer required before CICS writes an activity keypoint. A keypoint is a snapshot of inflight tasks in the system at that time.

During emergency restart, CICS needs to read back for records for only those tasks that are identified in a keypoint. CICS reads the system log backward until the first activity keypoint is encountered (which is the last activity keypoint taken).

Taking a keypoint imposes an overhead on the running system:

- If you set AKPFREQ too high, such that the keypoint frequency is too low, writing keypoints slows the system for only a short time.
- If you set AKPFREQ too low, such that the keypoint frequency is too high, the emergency restart time might be short, but you also incur increased processing, because more activity keypoints are processed.

It is advisable to set AKPFREQ to the default value of 4000. With an optimum setting of AKPFREQ, the whole of the system log can remain in the coupling facility.

Increasing the AKPFREQ value increases the amount of primary storage required for the system log. Decreasing the AKPFREQ value has the following effects:

- Restart time might be reduced.
- The amount of primary storage required for the system log decreases.

- Task wait time and processor cycles tend to increase.
- Paging might increase.

The last two effects can affect system performance, but not significantly.

If you set the AKPFREQ value to zero, emergency restart takes longer. In this situation, CICS cannot perform log tail deletion until shutdown, by which time the system log spills to auxiliary storage. Because there are no activity keypoints, CICS needs to read the whole of the system log, so it needs to retrieve the spilled system log from DASD offload data sets.

Activity keypoint frequency is determined by the AKPFREQ system initialization parameter. You can alter **AKPFREQ** while CICS is running by using the **CEMT SET SYSTEM AKP(value)** command.

The CICS log stream global statistics include information about the activity keypoint frequency. See “Logstream statistics” on page 611 for more information.

A message, DFHRM0205, is written to the CSMT transient data destination each time that a keypoint is taken.

AKPFREQ and MRO

In an MRO environment, the session allocation algorithm selects the lowest-numbered free session for use by the next task to run. Consequently, if many sessions have been defined (perhaps to cope with peak workload requirements), the higher-numbered sessions are less likely to be used frequently during quieter periods.

In an MRO environment, CICS implements the "implicit forget" process, an optimization of the two-phase commit. This means that when the mirror transaction at the remote end of an MRO connection completes any end-of-task processing, all information relating to the task is deleted when any new flow on that session arrives. This flow is usually the first flow for the next task or transaction allocated to run on the session as a result of the MRO session allocation algorithm.

Short-term variations in the arrival rate of transactions means that some mirror transactions waiting to process an implicit forget can persist for some time. This is particularly the case where such mirror transactions have been allocated to high-numbered sessions during a peak period, now passed, of transaction arrival rate.

The keypoint program uses an appreciable amount of processor capacity in processing persisting units of work such as those relating to mirror transactions waiting to process an implicit forget. This is exacerbated when the AKPFREQ value is low.

An optimum setting of AKPFREQ allows many of these persistent units of work to complete during normal transaction processing activity. This minimizes the processor processing used by the keypoint program. For this reason, you must be cautious when reducing the value of AKPFREQ to less than the default value.

The log defer interval (LGDFINT)

The **LGDFINT** system initialization parameter specifies the log defer interval used by CICS log manager when determining how long to delay a forced journal write request before starting the MVS system logger.

The value is specified in milliseconds. Performance evaluations of typical CICS transaction workloads have shown that a value of 5 milliseconds gives the best balance between response time and central processor cost.

CICS performance can be adversely affected by a change to the log defer interval value. Too high a value delays CICS transaction throughput due to the additional wait before starting the MVS system logger.

An example of a scenario where a reduction in the log defer interval might be beneficial to CICS transaction throughout would be where many forced log writes are being issued, and little concurrent task activity is occurring. Such tasks will spend considerable amounts of their elapsed time waiting for the log defer period to expire. In such a situation, there is limited advantage in delaying a call to the MVS system logger to write out a log buffer, since few other log records are added to the buffer during the delay period.

Although the range of possible values for the log defer interval is from 0 to 65535 milliseconds, the default of 5 milliseconds is considered to be the correct interval when setting the parameter in most cases.

A log defer interval value of less than 5 milliseconds reduces the delay in CICS log manager before starting the IXGWRITE macro. This might improve the transaction response time, but increases processor cost for the system because CICS has fewer journal requests into a given call to the MVS system logger, and so must start the IXGWRITE macro more often.

Conversely, increasing the log defer interval value to greater than 5 milliseconds increases the transaction response time, because CICS increases the delay period before starting the IXGWRITE macro. However, more transactions can write their own log data in to the same log buffer before it is written to the MVS system logger, and hence the total processor cost of driving IXGWRITE calls is reduced.

The log defer interval is determined by the **LGDFINT** system initialization parameter. **LGDFINT** can be altered with the **CEMT SET SYSTEM[LOGDEFER(value)]** command while CICS is running.

The CICS log stream global statistics capture information about the log defer interval. See "Logstream statistics" on page 611 for more information.

DASD-only logging

The primary storage used by a DASD-only log stream consists of a data space owned by the MVS logger and staging data sets. You can tune for DASD-only logging to improve performance.

No data is written to coupling facility structures. In its use of staging data sets, a DASD-only log stream is similar to a coupling facility log stream defined with DUPLEX(YES) COND(NO).

When the staging data set reaches its **HIGHOFFLOAD** limit, data is either deleted or offloaded until the **LOWOFFLOAD** limit is reached.

The following principles apply to DASD-only log streams as much as to coupling facility log streams:

- Size system logs so that system log data produced during the current activity keypoint interval, plus data not deleted at the previous activity keypoint, is retained in primary storage
- For the system log, avoid “staging data set full” conditions and offloading to auxiliary storage.

The basic principles of sizing the staging data set for a DASD-only log stream are the same as for sizing a staging data set for a coupling facility log stream, as described in “Tuning the size of staging data sets” on page 235. Take the values that you obtain as a starting point, and monitor your logger environment to adjust the size of the staging data set.

Use the following formula to calculate a starting point for the size of the staging data set for the system log. The formula calculates the value to be specified on the **STG_SIZE** parameter of the log stream definition; that is, the size is expressed as a number of 4 KB blocks.

```
Staging
DS size [No. of 4K blocks] = (AKP duration) * No. of log writes per second
                                               for system log
where:
AKP duration = (CICS TS 390 AKPFREQ) / (No. of buffer puts per second)
```

The values for the number of log writes per second and buffer puts per second can be taken from your CICS statistics. In CICS Transaction Server releases, the log stream statistics fields collect these statistics as "write requests" (LGSWRITES) and "buffer appends" (LGSBUFAPP), and you can divide the totals by the number of seconds in your statistics interval.

If you want to make a more accurate estimate for the size of the staging data set, consult the following documents:

- The IBM Redpaper Performance Considerations and Measurements for CICS and System Logger, REDP-3768. This document supplies guidance on the interactions between CICS and z/OS System Logger, provides examples of different CICS and System Logger configurations, and demonstrates the tuning process.
- The IBM Redbooks publication Systems Programmer's Guide to: z/OS System Logger, SG24-6898. This document explains how to obtain and use an IXGRPT1 report to estimate the size of a staging data set for a DASD-only log stream. (IXGRPT1 is a sample program provided with z/OS.)

Chapter 17. CICS temporary storage: Performance and tuning

CICS temporary storage is intended for short-lived data. An application can write data to temporary storage as a series of numbered items in a temporary storage queue. CICS also creates some temporary storage queues for its own use. Temporary storage is heavily used in many CICS systems.

The ways in which you can tune the use of CICS temporary storage depend on the locations of the temporary storage available to the CICS region. Temporary storage can be main storage in the CICS region, auxiliary storage in a VSAM data set, or shared temporary storage pools in a z/OS coupling facility. The temporary storage can be associated with the local CICS region or a remote queue-owning region (QOR). For an overview of the locations for temporary storage, see “CICS temporary storage: overview” on page 242.

For main temporary storage, you can monitor the use of storage and use the **TSMAINLIMIT** system initialization parameter to set a suitable limit. For more information about tuning main temporary storage, see “Main temporary storage: monitoring and tuning” on page 244.

For auxiliary temporary storage, you must balance several factors when you set up the VSAM data set and when you are tuning the use of CICS temporary storage. The following factors affect the performance of auxiliary temporary storage:

- The control interval size for the data set
- The number of VSAM buffers in the CICS region
- The number of VSAM strings for I/O to the data set

For more information about tuning auxiliary temporary storage, see “Auxiliary temporary storage: monitoring and tuning” on page 246.

Consider setting up shared temporary storage pools to improve availability and support dynamic transaction routing. Shared temporary storage pools require temporary storage servers (typically one server in each z/OS image in the sysplex), but they have a number of advantages:

- No storage is used in the CICS region for the shared temporary storage pools.
- Shared temporary storage pools do not cause intertransaction affinities. Local temporary storage queues in main or auxiliary storage can cause intertransaction affinities, where affected transactions must run in the same region to access the queue. Intertransaction affinities can affect performance by limiting the scope for workload routing across AORs in a sysplex.
- Compared to remote queue-owning regions, access to temporary storage queues in shared temporary storage pools in a coupling facility is quicker.
- If you use more than one temporary storage server for each pool, availability is better than it is for a remote queue-owning region. If one temporary storage server or z/OS image fails, transactions can be dynamically routed to another application-owning region on a different z/OS image.

CICS temporary storage: overview

You can set up temporary storage for a CICS region in three locations: main storage, auxiliary storage, or shared temporary storage pools in a z/OS coupling facility.

Main storage
Main temporary storage is in 64-bit (above-the-bar) storage in the CICS region. You use the **TSMAINLIMIT** system initialization parameter to specify the amount of storage that is available to temporary storage queues.

You can use local main storage in the CICS region where the applications run, or you can function ship temporary storage requests to a remote queue-owning region (QOR).

Auxiliary storage
Auxiliary temporary storage is in a nonindexed VSAM data set named DFHTEMP. You define the available space and any additional extents when you set up this data set. Some 31-bit (above-the-line) storage is used in the CICS region for VSAM buffers to make control intervals available from the VSAM data set. You use the **TS** system initialization parameter to set the number of buffers. Like main temporary storage, auxiliary temporary storage can be associated with the local CICS region or a remote queue-owning region.

Shared temporary storage pools in a z/OS coupling facility
Shared temporary storage pools (TS pools) are in a z/OS coupling facility managed by a temporary storage data sharing server (TS server). Each pool corresponds to a list structure in the coupling facility. You specify the size of each temporary storage pool using the coupling facility resource manager (CFRM) policy definition utility in z/OS. Shared temporary storage pools do not use any storage in the CICS region, and applications access them directly from the local CICS region.

When applications use the WRITEQ TS and READQ TS commands to access temporary storage queues, the requests are processed by the CICS temporary storage domain, which creates temporary storage queues in the appropriate storage location and places the data in them. Any task can retrieve the data using the symbolic name of the temporary storage queue. The CICS temporary storage domain can process multiple requests concurrently, but it serializes requests made for the same temporary storage queue, and the queue is locked for the duration of each request.

You use TSMODEL resource definitions to set up models that CICS uses to create temporary storage queues. Each model specifies the following attributes for temporary storage queues with names that match the model:

- The location of the temporary storage where the queue must be stored
- Whether the temporary storage is associated with the local CICS region or a remote CICS region, such as a queue-owning region
- Whether the queue is deleted automatically by CICS, if it remains unused for a period of time and is not deleted by an application
- Whether the queue is recoverable

Table 22 on page 243 summarizes the storage usage and the features that you can select for temporary storage queues in each location.

Table 22. Features of temporary storage locations

Temporary storage location	Storage type	Automatic queue deletion	Recovery
Main storage	64-bit storage in CICS region	Available	Not available
Auxiliary storage	VSAM data set, plus 31-bit storage in CICS region for buffers	Available for non-recoverable queues	Available
Shared temporary storage pool	z/OS coupling facility	Not available	CICS recovery is not available, but the queues are persistent (they are not affected by a CICS restart)

CICS also creates some temporary storage queues for its own use. These queues can be in main temporary storage or auxiliary temporary storage. For example, CICS uses temporary storage for the following purposes:

- Basic mapping support (BMS) paging and routing
- Caching of messages
- Interval control
- The CICS execution diagnostic facility (EDF)
- Local queueing for MRO, ISC, and IPIC while the target system is unavailable

When you view the temporary storage queues in your CICS system, queues with names that start with these characters are CICS queues: **, $$, X'FA' through X'FF', CEBR, and DF.

Automatic deletion of temporary storage queues

CICS can automatically delete nonrecoverable temporary storage queues that have not been referenced recently. To use this feature, you set suitable expiry intervals in the temporary storage models (TSMODEL resource definitions).

Automatic deletion frees storage occupied by temporary storage queues that were not deleted by applications and that are no longer required.

The expiry interval for a temporary storage model applies to the temporary storage queues that are associated with that model. Temporary storage queues use the expiry interval that exists for the TSMODEL resource definition at the time that the queue is created.

By default, the expiry interval is zero, that is, no expiry interval applies to the temporary storage queues. Such queues are never eligible for automatic deletion.

You can set an expiry interval in minutes, up to a maximum of 900,000 minutes (that is 15,000 hours). CICS uses the value rounded up to the nearest multiple of 10 minutes. The interval count begins after each use of the temporary storage queue. If the queue is not used again before the expiry interval is reached, the queue becomes eligible for CICS to delete it automatically. When at least one nonzero expiry interval in at least one TSMODEL resource definition exists, CICS starts to scan the CICS region regularly to find eligible queues. The CICS clean up task scans the temporary storage queues in the CICS region and deletes the queues that are eligible for automatic deletion.

Expiry intervals apply to temporary storage queues in the following locations:
- Main temporary storage in the local CICS region.
- Nonrecoverable auxiliary temporary storage (DFHTEMP data set) associated with the local CICS region.
- Queues in shared temporary storage pools.

Expiry intervals do not apply to the following types of temporary storage queue, so CICS never deletes them automatically:
- Queues in auxiliary temporary storage that are defined as recoverable.
- Queues in a remote CICS region. To make CICS delete remote temporary storage queues, specify an expiry interval in a suitable TSMODEL resource definition in the region that owns the queues.
- Queues that CICS creates for its own use.
- Queues that do not match any temporary storage model.

If you change the expiry interval in a TSMODEL resource definition, existing temporary storage queues that match the model are not affected. Those queues continue to use the expiry interval that applied when they were created. If all the TSMODEL resource definitions with a nonzero expiry interval are deleted from a CICS region, CICS stops scanning for expired temporary storage queues.

When the CICS clean up task performs a scan, it issues message DFHTS1605. This message shows the number of temporary storage queues that were scanned and the number that were deleted. If the clean up task ends abnormally, it issues message DFHTS0001, and does not run again until CICS is restarted.

The CICS clean up task cannot delete temporary storage queues if the system has reached TSMAINLIMIT (see TSMAINLIMIT system initialization parameter in Reference -> System definition) and there was an attempt to write to a TS queue such that a TS request lock is held. In this situation, the DFHTS1605 message reports that 0 queues were deleted.

Automatic deletion for TST users

If your CICS region still uses a temporary storage table (TST), which can be used in combination with TSMODEL resource definitions, the TST might include a TSAGE parameter. TSAGE specifies an aging limit in days, up to 512 days, for temporary storage queues. If the TST includes a nonzero TSAGE and there is an emergency restart of CICS, CICS deletes temporary storage queues that were not referenced during the specified interval. The TSAGE parameter does not cause automatic deletion of queues at any other time.

Main temporary storage: monitoring and tuning

You can monitor and control the amount of storage in the CICS region that is used by temporary storage queues.

About this task

From CICS TS for z/OS, Version 5.1, main temporary storage is located in 64-bit storage, so the available space is greater than in earlier CICS releases. Main temporary storage does not require VSAM I/O activity or communication with a temporary storage server. However, temporary storage queues in main temporary storage are not recoverable.

The CICS temporary storage statistics show information about the use of main temporary storage. You can also use CICSPlex SM or CICS commands to see the amount of main temporary storage in use, and the current limit. When 75% or more of the maximum allowed storage is in use, CICS issues messages about this situation.

You use the **TSMAINLIMIT** system initialization parameter to specify the amount of storage in the CICS region that is available for temporary storage queues to use. You can specify an amount of storage in the range 1 - 32768 MB (32 GB).

However, you must also check the setting for the z/OS parameter **MEMLIMIT**. **MEMLIMIT** limits the amount of 64-bit storage that the CICS address space can use. Your setting for **TSMAINLIMIT** must not be greater than 25% of the **MEMLIMIT** value.

Procedure

1. Specify expiry intervals in your temporary storage models. When you specify an expiry interval, CICS can automatically delete temporary storage queues that match the models if they are not deleted by applications. For more information about expiry intervals, see Automatic deletion of temporary storage queues.
2. Use CICSPlex SM, CICS commands, or CICS statistics to monitor the amount of main temporary storage in use.
   - The CICS temporary storage global and summary statistics show the number of times that main temporary storage use reached the limit set by **TSMAINLIMIT**, and the peak amount of virtual storage that was used for data in main temporary storage.
   - The TEMPSTORAGE resource shows the storage in use compared to the maximum allowed limit.
3. Look out for messages from CICS about high usage of main temporary storage.
   - CICS issues message DFHTS1601 when 75% or more of the maximum allowed storage is in use.
   - CICS issues message DFHTS1602 if an application attempts to write an item of data that would make the main temporary storage in use exceed the maximum allowed limit (the **TSMAINLIMIT** value). In this situation, applications cannot write to temporary storage queues in main temporary storage until space becomes available.

   If either of these messages are issued, try to delete old temporary storage queues or increase the **TSMAINLIMIT** setting, as described in the following steps. CICS issues message DFHTS1604 when usage falls below 70% of the maximum allowed.
4. Before you change the **TSMAINLIMIT** setting, check your current setting for the z/OS parameter **MEMLIMIT**. The amount of storage that you make available for temporary storage queues must not be greater than 25% of the **MEMLIMIT** value. For information about the **MEMLIMIT** value for CICS and instructions to check the value of **MEMLIMIT** that currently applies to the CICS region, see "Estimating, checking, and setting MEMLIMIT" on page 100.
5. Optional: To change the amount of storage available for temporary storage queues, change the **TSMAINLIMIT** setting. You can change the **TSMAINLIMIT** setting in a running CICS system.
   - If you increase the **TSMAINLIMIT** setting and the new value is greater than 25% of the value of **MEMLIMIT**, **TSMAINLIMIT** remains unchanged and message DFHTS1607 is issued.

- If you decrease the **TSMAINLIMIT** setting, CICS attempts to maintain at least 25% free space in allowed storage above current utilization, so that temporary storage write requests do not reach the **TSMAINLIMIT** value too rapidly. The value is set as follows:
  – If there is currently less than 25% free space, **TSMAINLIMIT** remains unchanged. Message DFHTS1606 is issued.
  – If at least 25% of the new limit will be free space, the setting is decreased to the value that you choose.
  – If less than 25% of the new limit would be free space, setting is decreased to the current utilization plus 33% of that utilization.

  If the value of **TSMAINLIMIT** is changed, CICS issues message DFHTS1603, which shows the new setting.

Results

The following table shows the cost of main storage. In this example, *n* represents the number of items in the queue before it is deleted.

Table 23. The cost of main storage

WRITEQ	REWRITE	READQ	DELETEQ
1.0	0.8	0.8	0.71 + 0.23 x *n*

Auxiliary temporary storage: monitoring and tuning

The performance of auxiliary temporary storage is influenced by the characteristics of the VSAM data set DFHTEMP that you set up for temporary storage. It is also affected by the number of VSAM buffers and strings that you specify for the CICS region.

About this task

The CICS temporary storage statistics show information about the use of auxiliary temporary storage, the use of buffers and strings, and I/O activity. For additional information about data set performance, use RMF or the VSAM catalog.

The cost approximations for auxiliary TS queues do not include any VSAM I/O cost. A VSAM I/O costs approximately 11.5K instructions and occurs in the following situations:

- When attempting to write an item that does not fit in any buffer
- When reading an item that is not in the buffer
- When reading a control interval from DASD with no available buffer space, if the least recently used buffer must first be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/Os.

Procedure

The following actions can influence the performance of auxiliary temporary storage:

- You specify a control interval (CI) size when you set up the VSAM data set DFHTEMP. When the use of temporary storage by applications or by CICS changes in your CICS region, verify that the control interval size is still suitable.

If you write items larger than the control interval size to a temporary storage queue in auxiliary storage, CICS processes the items, but performance might degrade. For information about the control interval size, see The control interval size in the *CICS System Definition Guide*.

- For more efficient use of DASD space, you can specify secondary extents when you set up the VSAM data set DFHTEMP. CICS uses secondary extents if there are no control intervals remaining in DFHTEMP with sufficient space for new data. You can define a temporary storage data set with a primary extent large enough for normal activity, and with secondary extents for exceptional circumstances. For instructions to define additional extents, see Multiple extents and multiple volumes in the *CICS System Definition Guide*.
- To help ensure that space is not wasted in auxiliary temporary storage, specify expiry intervals in your temporary storage models for nonrecoverable queues. Expiry intervals make CICS automatically delete any temporary storage queues that might not be deleted by applications. When an unused queue is deleted from a DFHTEMP control interval, CICS can move the remaining records to the start of the control interval, and use the space for new data. Efficient deletion of old queues can reduce the time required to locate a control interval with free space, and reduce the need to use secondary extents. For more information about expiry intervals, see “Automatic deletion of temporary storage queues” on page 243.
- If you specify the system initialization parameter **SUBTSKS=1**, CICS runs temporary storage VSAM requests on the concurrent (CO) mode TCB, which could increase throughput.
- You use the **TS** system initialization parameter to specify the numbers of VSAM buffers and strings for auxiliary temporary storage in the CICS region. If auxiliary temporary storage is heavily used in the CICS region, you might want to experiment with adjusting these numbers. Increasing the numbers of buffers and strings can reduce task waits and VSAM I/O requests, but it also increases storage use in the CICS region.

Recoverable and nonrecoverable TS queues

The cost of temporary storage is different for the recoverable TS queue and the nonrecoverable TS queue.

The main difference between the cost of accessing recoverable and nonrecoverable TS queues is incurred at sync point time. For recoverable queues, the following events occur at sync point time:

- The VSAM I/O cost is incurred if any control interval has been used during the unit of work, and has not already reached DASD.
- The new DASD control interval addresses are put in the log buffer. The cost for recovery manager to do this is about 2000 instructions.
- A forced log write is requested and the sync point completes when the log buffer has been written to primary storage.

In each table, *n* represents the number of items in the queue before it is deleted.

Table 24. Recoverable TS queue

WRITEQ	REWRITE	READQ	DELETEQ
1.4	1.9	1.0	0.87 + 0.18 * *n*

Table 25. Nonrecoverable TS queue

WRITEQ	REWRITE	READQ	DELETEQ
1.3	1.8	1.0	0.75 + 0.18 * *n*

Chapter 18. CICS transient data (TD) facility: Performance and tuning

Transient data (TD) is used in many circumstances within CICS, and various options can affect the performance of this facility.

The circumstances in which transient data is used include:
- 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 transient data definitions.
- Passing extrapartition requests to the operating system access method for processing.

Limitations

Application requirements might dictate a lower trigger level, or physical or logical recovery, but these facilities increase processor requirements. Real and virtual storage requirements might be increased, particularly if several buffers are specified.

Implementation

Transient data performance is affected by the **TRIGGERLEVEL** and **RECOVSTATUS** operands in the transient data resource definitions that have been installed.

Recommendations

The following suggestions might help to reduce waits during QSAM processing:
- 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 that are 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 three blocks can be accommodated on each track.

Monitoring

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. CICS transient data statistics also show the peak size of each intrapartition transient data queue during the statistics interval. RMF or the VSAM catalog shows data set performance.

Recovery options

Recovery can affect the length of time for which a transient data record is enqueued.

You can specify one of three options:

- *No recovery*. If you specify no recovery, there is no logging, and no enqueuing for protecting resources.
- *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 might occur instantaneously. Records that have been written can be read by another task immediately. Control intervals (CIs) are released as soon as they have been exhausted. For every WRITEQ TD request, the CI buffer is written to the VSAM data set.

  Note: All other resources that offer recovery within CICS provide only logical recovery. Using backout in an abend situation would exclude your physically recoverable and nonrecoverable transient data from the backout.
- *Logical recovery*. Specify logical recovery when you want to restore the queues 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 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 "Using VSAM subtasking" on page 196).

Nonrecoverable TD queue

A nonrecoverable TD queue has costs associated with it.

WRITEQ	READQ	DELETEQ
1.5	1.3	1.3

Note:

The main difference between nonrecoverable and logically recoverable TD queues occurs at sync point time. At sync point, the new TD queue addresses are put in the log buffer and a forced log write is requested. The cost to put the data in the buffer is 2 K. The cost of writing the log buffer to the coupling facility is described in "Using coupling facility data tables" on page 199.

Logically recoverable TD queue

A logically recoverable TD queue has costs associated with it.

WRITEQ	READQ	DELETEQ
First: 2.8 Subsequent:1.5	First: 2.4 Subsequent:1.4	1.1

Notes:

The main difference between nonrecoverable and logically recoverable TD queues occurs at sync point time. At sync point, the new TD queue addresses are put in the log buffer and a forced log write is requested. The cost to put the data in the buffer is 2 K. The cost of writing the log buffer to the coupling facility is described in "Using coupling facility data tables" on page 199.

Physically recoverable TD queue

Physically recoverable WRITEQ requests involve forcing a VSAM I/O and forcing a log write to the coupling facility (CF) for every request.

WRITEQ	READQ	DELETEQ
19.7	First: 9.3 Subsequent:8.8	8.7

Intrapartition transient data considerations

The approximations for nonrecoverable and logically recoverable intrapartition transient data queues do not include any VSAM I/O cost.

A VSAM I/O operation costs approximately 11.5 K and occurs in the following situations:

- When attempting to write an item that will not fit in any buffer.
- When reading an item that is not in the buffer.
- When reading a control interval from DASD and there is no available buffer space. If this situation occurs, the least recently used buffer must first be written out. Therefore, under certain circumstances, a READQ could incur the cost of two VSAM I/O operations.

For more information about intrapartition transient data, see Intrapartition transient data.

Multiple VSAM buffers

When you use multiple buffers and strings for intrapartition transient data (TD) support, this can remove the possible constraint in transient data caused by the use of a single system-wide buffer (and string). You can use statistics 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 control intervals (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 TD 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 determine 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 the tuning process, there is a trade-off between improving transient data performance and increased storage requirements. Specifying a large number of buffers might decrease transient data I/O and improve concurrency, but might also 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 from the ECDSA during initialization.

Multiple VSAM strings

As far as concurrent input/output operations with CICS are concerned, the transient data (TD) programs issue VSAM requests whenever real input/output is required between the buffers and the VSAM TD 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 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 TD is specified by the **TD** system initialization parameter. The CICS default is 3.

Logical recovery

Logging and enqueuing occur with logical recovery transactions (including dynamic backout of the failing task's activity on the transient data queue). Logical recovery is generally 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 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 UOW. 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 UOW (sync point or task completion), sync point processing takes place and the queue entry is logged. Any purge requests are processed (during the UOW, a purge only marks the queue ready for purging). The empty control intervals are released for general transient data use. Any trigger levels reached during the UOW cause automatic task initiation to take place for those queues that have a trigger level greater than zero. The buffer is written out to the VSAM data set as necessary.

The DEQUEUE function on the queue 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 queue entry is logged after each READQ, WRITEQ, and DELETEQ command, and at an activity keypoint time (including the warm keypoint).

With *logical* recovery, the queue entry is logged at sync point and at activity keypoint time (including the warm keypoint).

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 where CICS uses the 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. A lengthy wait can occur for one of the following reasons:

- No buffer space is available.
- Secondary space is allocation.
- Volume (extent) switching is available.
- The data set has been opened or closed dynamically.
- A forced end of the volume has been caused by the application.
- The data set is defined on a physical printer and the printer has run out of paper.
- A RESERVE command has been issued for another data set on the same volume.

Therefore, 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 or CLOSE actions during peak periods.

An alternative method of implementing sequential data sets is to employ a CICS user journal. Table 26 summarizes the differences between these two methods.

Table 26. 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	2 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

The approximate calculations for performance costs in extrapartition TD queues do not include any I/O cost. An I/O operation for a physically sequential file costs approximately 7 K and occurs in the following situations:

- When attempting to write an item that does not fit in any buffer.
- When reading an item that is not in the buffer.
- When reading data from DASD and there is no available buffer space. If this situation occurs, the least recently used buffer must first be written out.

Therefore, under certain circumstances, a READQ could incur the cost of two I/O operations.

Extrapartition TD queues are nonrecoverable.

WRITEQ	**READQ**
1.2	1.0

Indirect destinations

To avoid specifying extrapartition data sets for the CICS-required entries (such as CSMT and CSSL) in CSD definitions for TD queues, 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.

Chapter 19. Global CICS enqueue and dequeue: Performance and tuning

Global CICS enqueue and dequeue extends the CICS application programming interface to provide an enqueue mechanism that serializes access to a named resource across a specified set of CICS regions contained within a sysplex.

Because global CICS enqueue and dequeue eliminates a significant cause of inter-transaction affinity, it enables better exploitation of parallel sysplex, providing better performance, capacity, and availability. It also reduces the need to provide inter-transaction affinity rules to dynamic routing mechanisms such as CICSPlex SM, thus reducing the system management cost of exploiting parallel sysplex.

CICS uses z/OS global resource serialization to provide sysplex-wide protection for the resources that participate in global CICS enqueue and dequeue. For more information on z/OS global resource serialization, see z/OS MVS Planning: Global Resource Serialization.

Implementation

You use an ENQMODEL resource definition to define each named resource for which the **EXEC CICS ENQ** and **EXEC CICS DEQ** commands have a sysplex-wide scope. The CICS regions that need to use sysplex-wide enqueue or dequeue function must all have the required ENQMODEL resources defined and installed. The recommended way to ensure this is for the CICS regions to share a CSD, and for the initialization group lists to include the same ENQMODEL groups. For more information on creating ENQMODEL resource definitions, see ENQMODEL resources in Reference -> System definition.

For applications where the resource name is configured dynamically, so is not known in advance, you can use the enqueue EXEC interface program exits XNQEREQ and XNQEREQC to supply characters at the start of the resource name that match a suitable ENQMODEL resource definition. For more information on these user exits, see XNQEREQ and XNQEREQC, enqueue EXEC interface program exits in Reference -> System programming.

When the **EXEC CICS ENQ** and **EXEC CICS DEQ** commands are issued for a resource, CICS checks for a matching installed ENQMODEL definition. If there is a matching ENQMODEL resource that specifies an enqueue scope, CICS passes the information to z/OS global resource serialization to manage the enqueue. z/OS global resource serialization provides sysplex-wide protection of the resource.

z/OS global resource serialization includes resource name lists (RNLs) that specify the scope of resources. RNL processing can cause the scope of resources to change from the scope that was specified in the ENQMODEL resource definition in CICS.

The default in z/OS is that global resource serialization searches the appropriate RNL for enqueue and dequeue requests, and uses the RNL to determine the scope of the resource. However, the default in CICS, as specified by the **NQRNL** system initialization parameter, is that all enqueue and dequeue requests specify **RNL=NO** and so are excluded from RNL processing. This action means that global resource serialization only uses the scope specified in the ENQMODEL resource definition in CICS, but it also means that the enqueue request is ignored by alternative serialization products, which impacts protection of the resource to systems outside the current global resource serialization environment that are using alternative serialization products. If you want z/OS global resource serialization to use RNL processing for enqueue and dequeue requests from CICS, specify the system initialization parameter **NQRNL=YES** for the CICS regions where RNL processing should be performed.

For more information on RNL processing for global resource serialization, see z/OS MVS Planning: Global Resource Serialization.

Recommendations

z/OS global resource serialization combines systems into a global resource serialization complex. One or more systems are connected to each other in a ring configuration (GRS=RING) or connected to a coupling facility lock structure in a star configuration (GRS=STAR). When global resource serialization is initialized as a star configuration, all the information about resource serialization is held in the ISGLOCK coupling facility structure. Global resource serialization accesses the coupling facility when a requestor issues an enqueue or dequeue instruction on a global names resource.

Note: Use GRS=RING with caution as this configuration can result in serious performance constraints. For performance reasons, in a sysplex of greater than two MVS images use a global resource serialization star configuration.

The performance impact can be for many reasons, but primarily it is due to the delay in having the request complete the ring. A large number of MVS images in the ring combined with a large value for RESMIL causes delays in the request completing the ring. The enqueue request cannot be granted until the request returns to the originating MVS image. Use a value of 0, or no greater than 1, for RESMIL in the GRSCNF member of SYS1.PARMLIB.

Chapter 20. CICS monitoring facility: Performance and tuning

The CICS monitoring facility collects data about the performance of all user-supplied and CICS-supplied transactions during online processing for later offline analysis. Monitoring data is useful for performance, tuning, and for charging your users for the resources they use. The records produced by CICS monitoring are of the MVS System Management type 110 and they are written to an SMF data set.

Chapter 28, "Collecting and processing data for CICS monitoring," on page 295 has information about the different types of monitoring data.

In terms of performance, collecting performance class data can be a significant overhead. The overhead is likely to be about 5% to 10%, but depends on the workload. MVS address space or RMF data can be gathered whether or not the CICS monitoring facility is active, to give an indication of the performance overhead incurred when using the CICS monitoring facility. CICS Monitoring Domain statistics show the number of monitoring records produced of each type.

If you do not need accounting information because other billing processes exist, and you have other means of gathering any performance data required, do not use the CICS monitoring facility to collect performance class data. Do not collect exception class data if you do not require it.

Recording of monitoring data incurs an overhead, but, to tune a system, both performance and exception information might be required. If tuning is not a daily process, the CICS monitoring facility might not need to be run all the time. When tuning, run the CICS monitoring facility during peak volume times because at those times performance problems typically occur.

To help reduce the overhead, data compression for monitoring records is set as the default. If overuse of the SMF data set is a potential problem, consider excluding fields from monitoring records.

"Controlling CICS monitoring" on page 302 explains how to set CICS monitoring facility options using system initialization parameters and how to change these options while CICS is running.

Chapter 21. CICS trace: performance and tuning

The CICS tracing, handled by the CICS trace domain, records all requests that application programs make to CICS for various services. The storage and processing requirements depend on the number of trace entries that are recorded. Using CICS trace increases processing requirements considerably. Not using CICS trace, however, reduces the amount of problem determination information that is available for the CICS region.

CICS does not provide a direct measurement of processor use caused by tracing. RMF can show the processing and storage requirements. Auxiliary trace, where trace entries are written to auxiliary storage, has an additional cost because of the I/O operations. Although two buffers are used for auxiliary trace, even if the I/O can be overlapped, the I/O rate is quite large for a busy system.

You can control the amount of tracing that is done in a CICS region. You can limit the transactions or components that are traced, and the levels of trace data that are captured for them. You can set these options at CICS startup by using CICS system initialization parameters, or while CICS is running by using CICS interfaces. For information about defining the tracing that is done in the CICS region, see Using traces in problem determination in Troubleshooting.

CICS always performs exception tracing when it detects an exception condition, so you always have first failure data capture regardless of the limits that you set for CICS trace. In a production region, for example, you might want to set tracing options so that exception traces are written to auxiliary storage, but no other tracing is carried out. For instructions describing how to do this, see CICS exception tracing in Troubleshooting.

The trace data produced by CICS trace has a number of possible destinations. Any combination of any of these destinations can be active at any time:
- The internal trace table
- The auxiliary trace data sets
- The MVS generalized trace facility (GTF) data sets
- The JVM server trace file in z/OS Unix System Services

Also, when a transaction dump is produced, CICS copies the internal trace table to produce the transaction dump trace table. For information about selecting trace destinations, see Setting trace destinations and tracing status in Troubleshooting.

Internal trace table: storage use

Every CICS region must always have an internal trace table. The internal trace table is used as a buffer for the other trace destinations. If no trace destinations at all are currently started, CICS still writes exception trace entries to the internal trace table to provide first failure data capture.

You use the **TRTABSZ** system initialization parameter to specify the size of the internal trace table at CICS startup. The minimum size of the internal trace table is 16 KB, and the maximum size is 1 GB. The default size is 12288 KB (12 MB). The trace table should be large enough to contain the entries needed for debugging purposes.

CICS obtains MVS 64-bit (above-the-bar) storage (outside the CICS DSAs) for the internal trace table.

If you change the size of the internal trace table, check your current setting for the z/OS parameter **MEMLIMIT**. **MEMLIMIT** limits the amount of 64-bit storage that the CICS address space can use. Your setting for **TRTABSZ** must remain within **MEMLIMIT**, and you must also allow for other use of 64-bit storage in the CICS region.

For information about the **MEMLIMIT** value for CICS, and instructions to check the value of **MEMLIMIT** that currently applies to the CICS region, see Estimating, checking, and setting MEMLIMIT in Improving performance. For further information about **MEMLIMIT** in z/OS, see Limiting the use of memory objects in the z/OS MVS Programming: Extended Addressability Guide.

Transaction dump trace table: storage use

When a transaction dump is produced, CICS copies the current internal trace table to produce the transaction dump trace table. CICS obtains MVS storage in 64-bit (above-the-bar) storage for the transaction dump trace table when a transaction dump is taken.

You use the **TRTRANSZ** system initialization parameter to specify the size of the transaction dump trace table. The minimum size is 16 KB, and the default size is 1024 KB.

Before CICS TS for z/OS, Version 4 Release 2, the transaction dump trace table was in 31-bit (above-the-line) storage. If you specified a small size for the transaction dump trace table at that time because of concerns about the availability of 31-bit storage, consider reviewing your **TRTRANSZ** value to provide a larger transaction dump trace table now that 64-bit storage is used.

Because the transaction dump trace table is in 64-bit storage, check your current setting for the z/OS parameter **MEMLIMIT** when you set the size of the trace table.

Auxiliary trace data sets: storage use

The auxiliary trace data sets are CICS-owned BSAM data sets on disk or tape. You must create the data sets before you start CICS; you cannot define them while CICS is running. For instructions to set up the auxiliary trace data sets, see Setting up auxiliary trace data sets in Configuring.

When you start auxiliary trace, either at CICS startup or while CICS is running, two 4 KB buffers for the CICS auxiliary trace data sets are allocated from MVS storage in the 31-bit (above-the-line) storage of the CICS region. MVS storage is not included in the CICS DSAs. The buffers are freed if you stop auxiliary trace, but they are not freed when you pause auxiliary trace or switch between the auxiliary trace data sets.

GTF data sets: storage use

The GTF buffer is allocated in 64-bit storage.

Chapter 22. CICS security: Performance and tuning

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 the authorization of an operator to run 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 increases both 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.

Implementation

Resource security is defined with the **RESSEC(YES)** attribute in the TRANSACTION definition. Command security is defined with the **CMDSEC(YES)** attribute in the TRANSACTION definition.

Monitoring

No direct measurement of the overhead of CICS security is given. RMF shows overall processor usage.

For more information, see RACF facilities in Securing.

Tuning for VERIFY TOKEN and SIGNON TOKEN

For best performance, ensure a sufficient number of open TCBs and define programs as threadsafe.

VERIFY TOKEN and **SIGNON TOKEN** requests run on an open TCB if possible. If the **VERIFY TOKEN** or **SIGNON TOKEN** is issued on an open TCB, it runs the request on this TCB. If the **VERIFY TOKEN** or **SIGNON TOKEN** is not issued on an open TCB, it switches to an open TCB if one is available, otherwise it switches to the Resource Owning (RO) TCB.

For best performance, set the **MAXOPENTCBS** system initialization parameter to a high enough value to allow sufficient open TCBs for the workload, and define programs that use **VERIFY TOKEN** or **SIGNON TOKEN** as threadsafe.

Chapter 23. CICS startup and shutdown time: Performance and tuning

If you want to reduce the amount of time required for CICS startup and normal shutdown, the areas to check include the startup procedures and autoinstall.

The IBM Redbooks publication, IBM System z® Mean Time to Recovery Best Practices, SG24-7816, contains information about how to customize CICS to minimize startup and shutdown time.

The following topics describe how to improve performance for CICS startup and shutdown.

Improving startup procedure

Because various configurations are possible with CICS, different aspects of the startup might require attention.

Procedure

You can define and tune aspects to improve startup performance. For more information about the CICS startup procedures and CICS system initialization, see .

1. Define the following items:
   a. The global and local catalogs
   b. The CICS system definition (CSD) data set
   c. The temporary storage data sets or transient data intrapartition data sets

   For details on how to define each data set, see Defining data sets in Configuring.
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. This might cause your system to go short on storage. Groups in the GRPLIST in the system initialization table (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 process minimizes the virtual storage that is tied up while CICS is processing the installation of the terminals.

   Note: All terminals are installed, even surrogate terminal control table (TCT) entries for MRO.
   You must ensure that the DFHVTAM group precedes any TERMINAL or TYPETERM definition in your GRPLIST. The DFHVTAM group is contained in the DFHLIST group list, so adding DFHLIST first to your GRPLIST ensures that the condition is met. If you do not add DFHLIST, the programs used to build the TCT are loaded for each terminal, thus slowing initial and cold starts.

   Do not have more than 100 entries in any group defined in the CSD. If you have too many entries, this might cause unnecessary overhead during processing, and make maintenance of the group more difficult.

3. Enure 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 might want to override can be coded in the **PARM** on the EXEC statement, or all of them can be overridden by specifying by specifying the ALL option for the **START** parameter.
4. If you do not intend to use CICS web support or the Secure Sockets Layer, ensure that TCPIP=NO is specified in the SIT. If TCPIP=YES is specified, the Sockets domain task control block is activated.
5. Tune the VSAM parameters of the local and global catalogs to suit your installation:
   a. Control interval (CI) sizes should be changed for optimum data and DASD sizes (see "Local shared resources (LSR) or nonshared resources (NSR)" on page 183 for more information). In most cases 2KB index CI, and 8 KB or 16 KB data CI, are suitable sizes.
   b. You can you specify the **BUFNI** and **BUFND** parameters in your JCL for the global catalog data set with the **AMP** parameter, rather than using **BUFSPACE**.
   c. 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)
   d. The number of index set records can then be computed. The 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 sequence set records: S = H / (D X C)*
      - *The number of index set records: I = T - S*

   Do not spend time trying to tune free space as it has no effect.

   You can obtain the number of index levels by using the IDCAMS LISTCAT command against a GCD after CICS has been shut down. Because a cold start mainly uses sequential processing, it should not require any extra buffers in addition to the buffers automatically allocated when CICS opens the file.
6. Consider whether to use the recovery manager utility program DFHRMUTL. On cold and initial starts, CICS normally deletes all the resource definition records from the global catalog. You can save the time taken to delete resource definition records by using the recovery manager utility program, DFHRMUTL. For more information, see Recovery manager utility (DFHRMUTL) in Reference -> Utilities.
   - Before a cold start, run DFHRMUTL with **SET_AUTO_START=AUTOCOLD,COLD_COPY** as input parameters. This creates a copy of the global catalog data set that contains only those records needed for a cold start. If the return code from this job step is normal, you can replace the original global catalog with the new copy (taking an archive of the original catalog if you want). An example of the JCL is provided with the description of DFHRMUTL.
   - Before an initial start, run DFHRMUTL with **SET_AUTO_START=AUTOINIT,COLD_COPY** as input parameters, and follow the same procedure to use the resulting catalog.
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, reducing the startup time.
9. Set the **RAPOOL** system initialization parameter to a value that allows faster autoinstall rates. For more information, see “Setting the size of the receive-any pool” on page 158.
10. Specify the buffer, string, and key length parameters in the LSR pool definition. Setting these parameters reduces the time taken to build the LSR pool, and also reduces the open time for the first file to use the pool.

    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.

    The use of DISP=(...,PASS) on any non-VSAM data set used in steps preceding CICS reduces allocation time the next time the data sets 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, which is a time-consuming process.

    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.

    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. Similar consideration should be applied to any dynamic LIBRARY resources installed at startup.You can use the shared modules in the link pack area (LPA) to help reduce the time required to load the CICS nucleus modules. For advice on how to install CICS modules in the LPA, see Installing CICS modules in the MVS link pack area in Installing.

    CICS does not load programs at startup time for resident programs. The storage area is reserved, but the program is loaded on the first access through program control for that program. This process speeds up the 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. If it is the first access, using the LOAD facility physically loads the program into its predefined storage location .

    The use of a program list table post initialization (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 might increase.

Autoinstall performance

You might want to increase the number of buffers to improve autoinstall performance. Increasing the number of buffers can stop the high-level index being read for each autoinstall.

If you have many terminals autoinstalled, shutdown can fail due to the value of the **MXT** system initialization parameter being reached or CICS becoming short on storage. To prevent this possible cause of shutdown failure, consider putting the CATD transaction in a class of its own to limit the number of concurrent CATD transactions. Also, the **AIQMAX** parameter can be specified to limit the number of devices that can be queued for autoinstall. This parameter protects against abnormal consumption of virtual storage by the autoinstall or delete process, caused as a result of some other abnormal event.

If the CATD transaction limit is reached, the **AIQMAX** system initialization parameter affects the LOGON, LOGOFF, and BIND processing by CICS. CICS requests the z/OS Communications Server to stop passing such requests to CICS. The z/OS Communications Server holds the requests until CICS indicates that it can accept further commands.

This occurs when CICS has processed a queued autoinstall request.

MVS automatic restart management

You can use the MVS automatic restart manager (ARM) to implement a sysplex-wide integrated automatic restart mechanism. A sysplex can use ARM and z/OS Communications Server persistent sessions spread across many terminal-owning regions (TORs) in a generic resource set.

Automatic restart management (ARM) is a sysplex-wide integrated restart mechanism that performs the following tasks:

- Restarts MVS subsystems in place if they abend (or if notified of a stall condition by a monitor program)
- Restarts all the elements of a workload (for example, CICS TORs, application-owning regions (AORs), file-owning regions (FORs), and DB2) on another MVS image after an MVS failure
- Restarts a failed MVS image

ARM and z/OS Communications Server persistent sessions provide good recovery times in the event of a TOR failure, and the TOR restart is reduced because only a fraction of the network must be rebuilt. You can log on to the generic resource while the failed TOR restarts.

ARM provides faster restart by providing surveillance and automatic restart. The need for operator-initiated restarts, or other automatic restart packages, are eliminated. For more information about MVS automatic restart management, see Implementing MVS automatic restart management in Installing and *z/OS MVS Setting Up a Sysplex*.

Chapter 24. CICS web support: performance and tuning

You can tune several aspects of your system in order to improve the performance of CICS web support.

See Performance and tuning of CICS Web support in the *CICS Internet Guide* for information and guidance about tuning CICS web support to maximize performance.

Chapter 25. CICS business transaction services: Performance and tuning

Business transaction services (BTS) introduced a business transaction model to CICS.

Effects

You can use BTS to create a type of program that controls the flow of many separate CICS transactions so that these individual transactions become a single business transaction.

Recommendations

A BTS transaction can comprise many separate CICS transactions and also can span a considerable execution time, so there are no specific performance recommendations for BTS transactions. However, some general observations can be useful.

Implementation

To support BTS functionality, CICS keeps data in new types of data sets: the local request queue (DFHLRQ) and a BTS repository. The local request queue data set stores pending BTS requests. Each CICS region has its own data set. The local request queue data set is a recoverable VSAM key-sequenced data set (KSDS). Tune it for best performance like a VSAM KSDS.

You can have one or more BTS repositories. A BTS repository is normally a VSAM KSDS and holds state data for processes, activities, containers, events, and timers. A BTS repository is associated with a process through the PROCESSTYPE definition. If the activities of a BTS process are to be dispatched on more than one CICS region, their BTS repositories must be shared between those regions. The repository can be either of the following file types:

- A VSAM KSDS file that is owned by a file-owning region and defined as REMOTE in participating regions
- A VSAM RLS file that is shared between the participating regions

To support the execution of the BTS processes, CICS runs one or many transactions. A BTS process consists of one or more activities. Each activity runs as a series of CICS transaction executions. If an activity becomes dormant, for example, it is waiting for an event, the activity restarts after that event occurs, and a new CICS transaction is started, even if this is a continuation of the business transaction. You might see many executions of the transaction identifier specified in a process or activity definition in the CICS statistics for a single BTS transaction. The application program that is run when an activity is executed is not necessarily the one that is defined in the transaction definition. In BTS, the Process or Activity definition in application programs can specify a different program to run.

The number of transactions run and the number and type of file accesses to the BTS repository, depend on how you choose to use BTS services. To see this information for your applications, examine the CICS statistics reports. Be aware that containers are stored in the BTS repository. Ensure that the repository is large enough to contain all the active BTS data. A good way to do this is to use scaling, based on a test system.

You can use monitor data, DFHCBTS, to collect information on activities within processes. For information about this data, see "Performance data in group DFHCBTS" on page 349.

For more information about business transaction services (BTS), see Overview of BTS in Product overview.

Chapter 26. Managing workloads

Workload management in a sysplex is provided by the z/OS Workload Manager (WLM) and by CICSPlex SM workload management.

The z/OS Workload Manager

The z/OS Workload Manager provides automatic and dynamic balancing of system resources (central processors and storage) across a sysplex.

The z/OS Workload Manager balances system resources by:
- Adopting a goal-oriented approach
- Gathering real time data from the subsystems that reflect performance at an individual task level
- Monitoring z/OS- and subsystem-level delays and waits that contribute to overall task execution times
- Dynamically managing the resources of the sysplex, using the performance goals, and the real time performance and delay data, as inputs to system resource management algorithms.

This resource management is particularly significant in a sysplex environment, but is also of value to subsystems running in a single z/OS image.

Note: If you use CICSPlex SM to control dynamic routing in a CICSplex, you can base its actions on the CICS response time goals of the CICS transactions as defined to the z/OS Workload Manager. See Dynamic routing with CICSPlex SM in *CICSPlex System Manager Managing Workloads*.

The z/OS Workload Manager provides the following benefits:
- Improved performance through z/OS resource management. Improvement can depend on many factors, for example:
  - System hardware configuration
  - How the system is partitioned
  - Whether CICS subsystems are single or multiregion
  - 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 z/OS sysplexes through improved overall capacity and increased work throughput.
- Simplified z/OS tuning. Systems that have an operating signature that makes it difficult or time consuming to attain or maintain optimal tuning by current means can benefit the most.

The main benefit is that you do not need to continually monitor and tune CICS to achieve optimum performance. You can set your workload objectives in the service definition, then the workload component of z/OS manages the resources and the workload to achieve your objectives.

The z/OS Workload Manager produces performance reports that you can use to establish reasonable performance goals and for capacity planning.

The CICS function for z/OS workload management incurs negligible impact on CICS storage.

CICS support for the z/OS Workload Manager is initialized automatically during CICS startup. All CICS regions (and other z/OS subsystems) running on a z/OS image with z/OS workload management are subject to the effects of the Workload Manager.

User-written resource managers and other non-CICS code that is attached to CICS through the RMI should be modified to provide z/OS Workload Manager support, if workload management is to work correctly for CICS-based tasks which cross the RMI into such areas.

The IBM Redbooks Publication System Programmer's Guide to: Workload Manager, SG24-6472-03, gives a broad understanding of the Workload Manager component of the z/OS system. It covers basic aspects of WLM together with the new functions available in the z/OS release up to z/OS 1.7. The book provides a discussion on how to create WLM policies based on business goals and the types of transactions you run in your systems.

Terms used in z/OS workload management

The following terms are used in the description of z/OS workload management.

classification rule
A rule used by the workload manager component of z/OS to assign a service class.

service class
A group of work that has the same service goals or performance objectives, resource requirements, or availability requirements. For workload management, a service goal and, optionally, a resource group is assigned 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 z/OS images using z/OS workload management 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
A group of service classes.

Span of z/OS Workload Manager operation

The z/OS Workload Manager operates across a sysplex. There can be only one active service policy for all z/OS images running in a sysplex.

All CICS regions (and other z/OS subsystems) running on a z/OS image with z/OS workload management active are subject to the effects of workload management.

If the CICS workload involves non-CICS resource managers, such as DB2 and DBCTL, CICS passes information through the resource manager interface (RMI) to enable the z/OS Workload Manager to relate the part of the workload within the non-CICS resource managers to the part of the workload within CICS.

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.

Performance goals for CICS regions

You can define performance goals, such as response times, for CICS (and other z/OS subsystems that comprise your workload).

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.

To define the performance goals for CICS regions, allocate each CICS job a service class and then specify target response times for the service class. Typically, production regions and test regions are placed in different service classes, because response times for production regions are more critical than for test regions.

Workload management also collects performance and delay data, which can be used by reporting and monitoring products, such as the Resource Measurement Facility (RMF), Tivoli Decision Support for z/OS, or vendor products.

The service level administrator defines your installation's performance goals, and monitoring data, 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).

Defining classification rules for your CICS workload

Classification 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 z/OS subsystem type that uses workload management. The work qualifiers that CICS can use (and which identify CICS work requests to the z/OS Workload Manager) are:

LU LU name

LUG LU name group

SI Subsystem instance (generic applid)

SIG Subsystem instance group

TN Transaction identifier

TNG Transaction identifier group

UI Userid

UIG Userid group.

Note:

1. Typically, work is classified in the region in which it arrives in CICS. For example, work originating from a user terminal is typically classified in a terminal-owning region. Web requests are typically classified in a listener region. Work originating in an application-owning region is classified in that region. Where a work request is passed between CICS regions, the transaction is not reclassified in each region. Instead, the original classification is passed with the transaction from region to region.
2. You can use group qualifiers to specify groups of transaction IDs or user IDs; for example, GRPACICS could specify a group of CICS transaction IDs, which you could specify in classification rules by TNG GRPACICS. Using group qualifiers is a much better method of specifying classification rules than classifying 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, the z/OS 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.

Defining service classes

Service classes 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.

Note:
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 z/OS workload management cannot find a matching service class in the classification rules—for example, if the couple data set becomes unavailable.

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.

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 z/OS Workload Manager first, and observe the effect on CICS performance. Once the z/OS 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.

  You should take care when choosing the priority to assign to each transaction. Although you can specify transaction priorities from 1 to 255, you should avoid using a large number of closely spaced values. You will get as much benefit if you use a small number of widely spaced values.

  The priority assigned by the CICS dispatcher must be compatible with the performance parameters defined to the z/OS Workload Manager.
- Maximum number of concurrent user tasks for the CICS region.

- Maximum number of concurrent tasks in each transaction class.
- Maximum number of sessions between CICS regions.

CICSPlex SM workload management

CICSPlex SM workload management directs work requests to a target region that is selected using one of four routing algorithms.

The queue algorithm (QUEUE)
CICSPlex SM routes work requests initiated in the requesting region to the most suitable target region in the designated set of target regions.

The link neutral queue algorithm (LNQUEUE)
The link neutral queue algorithm corresponds to the queue algorithm, except that the type of connection between the routing and target region is not considered.

The goal algorithm (GOAL)
CICSPlex SM routes work requests to the target region that is best able to meet the goals that have been predefined using the z/OS Workload Manager.

The link neutral goal algorithm (LNGOAL)
The link neutral goal algorithm corresponds to the goal algorithm, except that the type of connection between the routing and target region is not considered.

For more information, see Workload routing.

The CICSPlex SM dynamic routing program EYU9XLOP is invoked to route work requests to the selected target region. EYU9XLOP supports both workload routing and workload separation. You define to CICSPlex SM which requesting, routing, and target regions in the CICSplex can participate in dynamic routing, and any affinities that govern the target regions to which particular work requests must be routed. The output from the CICS Interdependency Analyzer can be used directly by CICSPlex SM. For information about the CICS Interdependency Analyzer, see the CICS Interdependency Analyzer for z/OS User's Guide and Reference, and the IBM Redbooks publication IBM CICS Interdependency Analyzer.

There are no special requirements for using CICSPlex SM workload management, which supports both the distributed routing and dynamic routing models of CICS. Workload management of the following types of requests is supported:

- Dynamic transaction routing
- Dynamic DPL
- Start requests
- BTS activities
- 3270 link requests

CICSPlex SM workload management offers the following benefits:

- A dynamic routing program to make more intelligent routing decisions; for example, based on workload goals.
- Improved CICS support for z/OS goal-oriented workload management.
- Easier access to a global temporary storage owning region in the z/OS sysplex environment. This avoids intertransaction affinity that can occur with the use of local temporary storage queues.

- Intelligent routing (through CICSPlex SM) in a CICSPlex that has at least one requesting region linked to multiple target regions.

For information about setting up and using CICSPlex SM workload management, see Managing workloads in *CICSPlex System Manager Concepts and Planning* and Introduction to workload management in *CICSPlex System Manager Managing Workloads*.

Chapter 27. Monitoring using RMF

You can use the CICS monitoring facility with the Resource Measurement Facility (RMF) to perform 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.

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.

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.

Terms used in RMF reports

It might help to relate some of the terms used in an RMF workload activity report to the more familiar CICS 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*.

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*.

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 or CSNE waiting for work.

These user tasks are shown 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*.

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, 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, temporary 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

An RMF workload activity report contains “snapshot data” which is data collected over a relatively short interval. The RMF reports provided in this section are examples of possible data that might be reported for CICS and IMS in an RMF workload activity report and some possible explanations for the data.

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 inconsistency 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. Figure 22 on page 284 illustrates this inconsistency.

For example, an AOR can finish processing transactions, the completion of which are included in the current reporting interval, while the TOR might not complete its processing of the same transactions during the same interval.

Figure 23 shows an example of the work manager state section for a service class

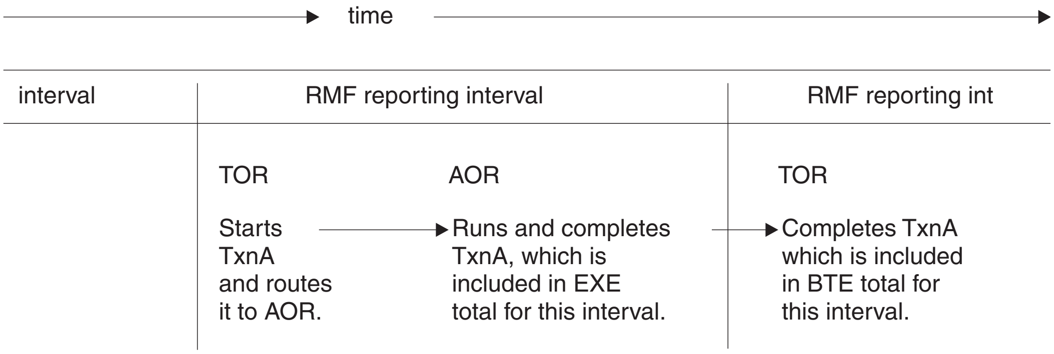


Figure 22. Illustration of snapshot principle for RMF reporting intervals

representing CICS transactions accessing DBCTL.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSHR RESOURCE GROUP=*NONE PERIOD=1 IMPORTANCE=1

-TRANSACTIONS--  TRANSACTION TIME   HHH.MM.SS.TTT
AVG       0.00   ACTUAL                      114
MPL       0.00   EXECUTION                    78
 ENDED         216  QUEUED                     36
END/S     0.24  R/S AFFINITY               0
#SWAPS       0  INELIGIBLE                 0
 EXECTD        216  CONVERSION                 0
AVG ENC   0.00  STD DEV                   270
REM ENC   0.00
MS ENC    0.00
          RESP  --------------------------STATE SAMPLES BREAKDOWN (%) -----------------------  ----STATE------
SUB   P   TIME  --ACTIVE-- READY IDLE  ----------------------WAITING FOR---------------------  SWITCHED SAMPL (%)
TYPE       (%)   SUB  APPL             CONV PROD                                              LOCAL SYSPL  REMOT
CICS BTE  93.4  10.9   0.0   0.0  0.0  89.2  0.0                                               89.2   0.0    0.0
CICS EXE  67.0  19.7   0.0  10.6  0.0  0.0  69.7                                                0.0   0.0    0.0
```

Figure 23. Hotel reservations service class

The fields in this RMF report describe an example CICS hotel reservations service class (CICSHR). CICS transactions have two phases:

- The *begin-to-end phase* (CICS BTE) takes place in the first CICS region to begin processing a transaction. Typically this region is a terminal-owning region (TOR). The TOR is responsible for starting and ending the transaction.
  - The **ENDED** field shows that 216 hotel reservation transactions completed.
  - The **ACTUAL** time shows that the 216 transactions completed in an average transaction time of 0.114 seconds.
- The *execution phase* (CICS EXE) can take place in an application-owning region (AOR) and a resource-owning region such as an FOR. In this example, the 216 transactions were routed by a TOR to an AOR.
  - The **EXCTD** field shows that the AORs completed 216 transactions in the interval.
  - The **EXECUTION** time shows that on average it took 0.078 seconds for the AORs to run the 216 transactions. The **EXECUTION** time applies only to the **EXCTD** transactions.

Begin-to-end phase analysis

While running these transactions, CICS records the states the transactions are experiencing. RMF reports the states in the `STATE SAMPLES BREAKDOWN (%)` section of the report, with one line for the begin-to-end phase, and another for the execution phase. Because there is a `CICS BTE` and `CICS EXE` field, you can assume that the time spent in the TOR represents the BTE phase and the time spend in the AOR represents the EXE phase. There is one EXE phase summarizing all the time spend in one or more AORs.

The CICS BTE total field shows that the TORs have information covering 93.4% of the response time, the analysis of which is shown in the remainder of the row. RMF does not have information covering 100% of the 0.114 seconds response time, because it take some time for the system to recognize and assign incoming work to a service class before it can collect information about it.

For most of the 93.4% of the time, the transactions did not run in the TOR, but had been routed locally to an AOR on the same MVS image. You can see this by the `SWITCHED SAMPL (%) LOCAL` field, which is 89.2% of the total state samples. This value accounts for 83.3% of the response time, because 100% of the total state samples correspond to 93.4% of the response time (89.2 x 93.4 / 100 = 83.3%). This value of 89.2% is close, if not equal, to the `WAITING FOR CONV` field. which indicates that there is no delay in the TOR once the AOR has returned the transactions.

Execution phase analysis

The total execution time is some percentage of the total response time. It is the `EXECUTION` transaction time (0.078), divided by `ACTUAL` transaction time (0.114), which is 68.4%. The CICS execution phase (`CICS EXE` field) covers 67% of the response time. Some of that time the work is active in the AOR, sometimes it is waiting behind another task in the region, but 69.7% of the total state samples in the `PROD` field (which corresponds to 69.7 x 67 / 100 = 46.7% of the response time) were found outside of CICS, waiting for another product to provide some service to these transactions. Based on the configuration of the system, the transactions are accessing DBCTL.

The `LOCL`, `SYSP`, and `REMT` state percentages appear in the `WAITING FOR` section if greater than zero and show the percentages of the total state samples the service class was delayed in the these states when CICS was waiting to establish a session. The `STATE SWITCHED SAMPL (%)` fields `LOCL`, `SYSPL`, and `REMOT` show the percentage of the state samples in which transactions were routed using MRO, MRO/XCP, or z/OS Communications Server connections.

RMF report example: very large response time percentage

The following report shows an example of a work manager state section for the CICSPROD service class.

In column `RESP TIME (%)`, both the `CICS EXE` and the `CICS BTE` rows show inflated percentages: 78.8K and 140.

```
REPORT BY: POLICY=HPTSPOL1 WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=1

TRANSACTIONS    TRANS.-TIME   HHH.MM.SS.TTT
AVG      0.00   ACTUAL                  111
MPL      0.00   EXECUTION               123
ENDED    1648   QUEUED                    0
END/S    1.83   R/S AFFINITY              0
#SWAPS      0   INELIGIBLE                0
EXCTD    1009   CONVERSION                0
AVG ENC  0.00   STD DEV                 351
REM ENC  0.00
MS ENC   0.00

           RESP ---------------------------- STATE SAMPLES BREAKDOWN (%)----------------------   ------STATE------
SUB    P   TIME  --ACTIVE--  READY  IDLE   -------------------------WAITING FOR---------------   SWITCHED SAMPL(%)
TYPE        (%)   SUB  APPL                 MISC   PROD  CONV  I/O                                LOCAL SYSPL REMOT
CICS  BTE  78.8K  0.2   0.0    0.3   2.5   96.7    0.0   0.3  0.0                                  0.3   0.0   0.0
CICS  EXE   140  65.6   0.0    2.2   0.0    0.0   32.4   0.0  0.1                                  0.0   0.0   0.0
```

Figure 24. Response time percentages greater than 100

Possible explanations

Long-running transactions

The report shows how long-running transactions can inflate the value for RESP TIME (%). While the following example does not explain the exact values in the figure, it explains why this inflation is possible.

Suppose 100 transactions have ended within 1 second, and one transaction has been running for 5 minutes and is still executing when the RMF interval expires. The ACTUAL transaction time shows an average response time of 1 second, and RMF shows the breakdown into the states recorded by CICS. The subsystem, however, recorded a total of 6 minutes and 40 seconds (5 minutes plus 100 seconds) worth of data. That is an average of 4 seconds worth of data for each completed transaction, which is 4 times the 1 second response time. The state samples breakdown, however, shows information representing 100% of the state samples.

Also, when the long-running transaction completes, it could easily distort the average response time during that interval. The RMF standard deviation and distribution of response times emphasizes 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 any AOR. Long-running routed transactions could result in many samples of WAITING FOR CONV (waiting for conversion) in the CICS BTE phase, as well as states recorded from the AOR in the execution phase.

Long-running transactions that are not routed execute completely in a TOR and have no CICS EXE phase data could inflate any state 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 example, these transactions could include the IBM reserved transactions such as CSNC and CSSY or customer defined transactions. Never-ending transactions are reported similarly to long-running transactions. However, for never-ending CICS transactions, RMF might report high percentages in the IDLE, WAITING FOR TIME, or the WAITING FOR MISC fields.

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 response time, so you might see percentages close to 100% in the IDLE state for completed transactions.

Service class includes dissimilar work

A service class that mixes customer and IBM transactions, short and long or never-ending transactions, routed and non-routed transactions, or 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 could be true for both IMS and CICS and can be expected if the service class is the subsystem default service class. The default service class is defined in the classification rules. It is the service class to which all work in a subsystem is assigned that is not assigned to any other service class.

Possible actions

Group similar work into service classes

Make sure your service classes represent a group of similar work. You could create additional classes, although you are recommended to create only a small number of service classes for CICS work. If there are transaction for which you want the RMF state samples 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, understand that the response time percentage could include long-running or never-ending transactions. RMF data for such service classes might not make immediate sense.

RMF report example: response time breakdown data is all zero

The following report shows an example of a work manager state section for the CICSLONG service class.

The RESP TIME (%) field shows a 0.0 value.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSLONG  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=1

                                                           CICS Long Running Internal Trxs

-TRANSACTIONS--  TRANS.-TIME   HHH.MM.SS.TTT.SS.TTT
AVG        0.00  ACTUAL                        0
MPL        0.00  EXECUTION                     0
ENDED         0  QUEUED                        0
END/S      0.00  R/S AFFINITY                  0
#SWAPS        0  INELIGIBLE                    0
EXCTD         0  CONVERSION                    0
AVG ENC    0.00  STD DEV                       0
REM ENC    0.00
MS ENC     0.00

           RESP  -------------------------------STATE SAMPLES BREAKDOWN (%)---------------   ---------STATE---
SUB    P   TIME  --ACTIVE-- READY IDLE  ----------------------------WAITING FOR----------------   SWITCHED SAMPL (%)
TYPE        (%)   SUB  APPL             CONV  I/O  PROD  DIST  REMT  LOCK                          LOCAL SYSPL REMOT
CICS  BTE   0.0  70.8   0.0   1.4  0.7  11.2  9.2   0.3   5.3   1.2   0.0                           11.2   0.0   0.0
CICS  EXE   0.0  43.2   0.0   0.2  0.1  31.8  10.4  8.7   0.0   2.9   2.8                            0.0   0.0   0.0
```

Figure 25. Response time breakdown percentages all 0.0

Possible explanations

No transactions completed

While a long-running or never-ending transaction is being processed, RMF stores the service class state samples in SMF 72.3 records. But when no transactions have completed, the average response time is 0. However, the calculations for the state samples breakdown will result in values greater than 0.

RMF did not receive data from all systems in the sysplex
The postprocessor might have been given SMF records from only a subset of the systems running in the sysplex. The report might 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, on this MVS image.

Possible actions

Do nothing
You might have created this service class to prevent the state samples of long-running transactions from distorting data for your production work.

Combine all SMF records for the sysplex
When a single MVS image that does not have TORs is combined with another MVS image that does have TORs and therefore does report response times, the states and response time from the first image will be combined by RMF with the states and response time from the second.

RMF report example: execution time is greater than response time

The following report shows an example of a work manager state section for the CICSPROD service class.

In the example, there is a response time of .091 seconds, but an execution time of .113 seconds. The example also shows 1731 ENDED transactions, yet the EXCTD field shows that only 1086 transactions have been executed.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD  SERVICE CLASS=CICSPROD RESOURCE GROUP=*NONE   PERIOD=1 IMPORTANCE=1

-TRANSACTIONS--   TRANS.-TIME   HHH.MM.SS.TTT
AVG        0.00   ACTUAL                   91
MPL        0.00   EXECUTION               113
ENDED      1731   QUEUED                    0
END/S      1.92   R/S AFFINITY              0
#SWAPS        0   INELIGIBLE                0
EXCTD      1086   CONVERSION                0
AVG ENC    0.00   STD DEV                  92
REM ENC    0.00
MS ENC     0.00
```

Figure 26. Execution time greater than response time

Possible explanations

Mixed routed and non-routed CICS transactions
The AORs might have recorded states which account for more time than the average response time of all the transactions. The non-routed transactions do not show up in the EXE phase. In addition, most non-routed transactions end very quickly and decrease the actual response time. 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 transactions they participated in.

Possible actions

Classify routed and non-routed transactions to different service classes
If you classify routed and non-routed transactions separately, the numbers will be consistent with the expectation.

RMF report example: large SWITCH percentage in CICS execution phase

The following report shows a work manager state data section for a CICSPROD service class.

The LOCAL value in the SWITCHED SAMPLE(%) section shows a value of 7092.

```
REPORT BY: POLICY=HPTSPOL1  WORKLOAD=PRODWKLD SERVICE CLASS=CICSPROD  RESOURCE GROUP=*NONE  PERIOD=1 IMPORTANCE=1

 -TRANSACTIONS--   TRANS.-TIME   HHH.MM.SS.TTT
 AVG        0.00   ACTUAL                  150
 MPL        0.00   EXECUTION               134
 ENDED      3599   ACTUAL                   16
 END/S      4.00   R/S AFFINITY              0
 #SWAPS        0   INELIGIBLE                0
 EXECUTD    2961   CONVERSION                0
 AVG ENC    0.00   STD DEV                 446
 REM ENC    0.00
 MC ENC     0.00

            RESP  ---------------------------RESPONSE TIME BREAKDOWN IN PERCENTAGE--------------   ------STATE------
 SUB    P   TIME  --ACTIVE-- READY  IDLE  ---------------------------WAITING FOR----------------   SWITCHED TIME (%)
 TYPE        (%)   SUB  APPL                MISC  PROD  CONV  I/O                                  LOCAL SYSPL REMOT
 CICS  BTE  26.8K  0.3   0.0   0.4   2.5  96.3   0.0   0.6  0.0                                      0.6   0.0   0.0
 CICS  EXE  93.7  41.2   0.0   6.0   0.0   0.0  52.7   0.0  0.0                                     7092   0.0   0.0
```

Figure 27. Large SWITCH precentage in a CICS execution environment

Possible explanations

Distributed transaction processing
If a program initiates distributed transaction processing to multiple back-end regions, there can be many AORs all associated with the original transaction. each of these multiple back-end regions can indicate they are switching control back to the front-end region (SWITCH to another region on the LOCAL MVS image, or to a region on another MVS image in the sysplex). Thus, with a one-to-many mapping like this, there are many samples of the execution phase of requests switched - long enough to exceed 100% of the response time of other work completing in the service class.

Distributed program link (DPL)
The distributed program link function builds on the distributed transaction functions available in CICS by enabling a CICS program (the client program) to call another program (the server program) in another CICS region.

While the service program is running, the client program will reflect that it is switched to another CICS region.

There are no possible actions.

RMF report 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 over a few days.

Possible explanations

Conversion from ISC link to MRO

When two CICS regions are connected using an intersystem communication (ISC) link, they behave differently than when they are connected using multi-region operation (MRO). One key difference is that, with ISC, both the TOR and the AOR are receiving a request from z/OS Communications Server, so each believes it is starting and ending a given transaction.

- If a user request is routed from the TOR to an AOR using ISC, two transactions complete. Assuming they have response times of 1 second and 0.75 seconds, the resulting average is 0.875 seconds.
- If a user request is routed from the TOR to an AOR using MRO, the TOR transaction completes taking 1 second. The AOR reports the 0.75 seconds as execution time.

Therefore, converting from an ISC link to an MRO connection for the same workload, as shown in this example, could result in half the number of ended transactions and in increase in the response time reported by RMF. The difference could be much more significant if the AOR to FOR link was converted.

Possible actions

Increase CICS transaction goals

Increase the CICS transaction goals before converting from ISC to MRO, particularly if the FOR transactions are classified to the same service class as your end-user transactions.

An explanation of the difference between a DFHSTUP transaction report and an RMF workload report

CICS transaction manager global statistics include all transactions in all regions in the interval or summary reports from DFHSTUP, but the z/OS WLM workload activity report includes only the transactions in Begin-To-End (BTE) phase and execution (EXE) phase. For WLM reporting purposes, the execution phase applies to only routed transactions in the AOR.

Figure 28 on page 291 shows the significance of the difference between the performance reports created for the region by DFHSTUP, and those generated by the RMF workload activity report for the reporting performance group number (RPGN). If you are not familiar with the RMF workload activity report, see "Interpreting the RMF workload activity data" on page 283 for more information.

In the terminal-owning region (TOR), the WLM reports for a given transaction are included in the RMF workload activity reports for the RPGN defined for the service class (for example, CICSPROD). In the application-owning region (AOR), notifications for routed transactions are included in the RMF workload activity reports when reporting execution phases in the CICS AOR. Transaction WLM notifications for mirror transactions are ignored by the z/OS WLM when reporting execution phases in the CICS FOR.

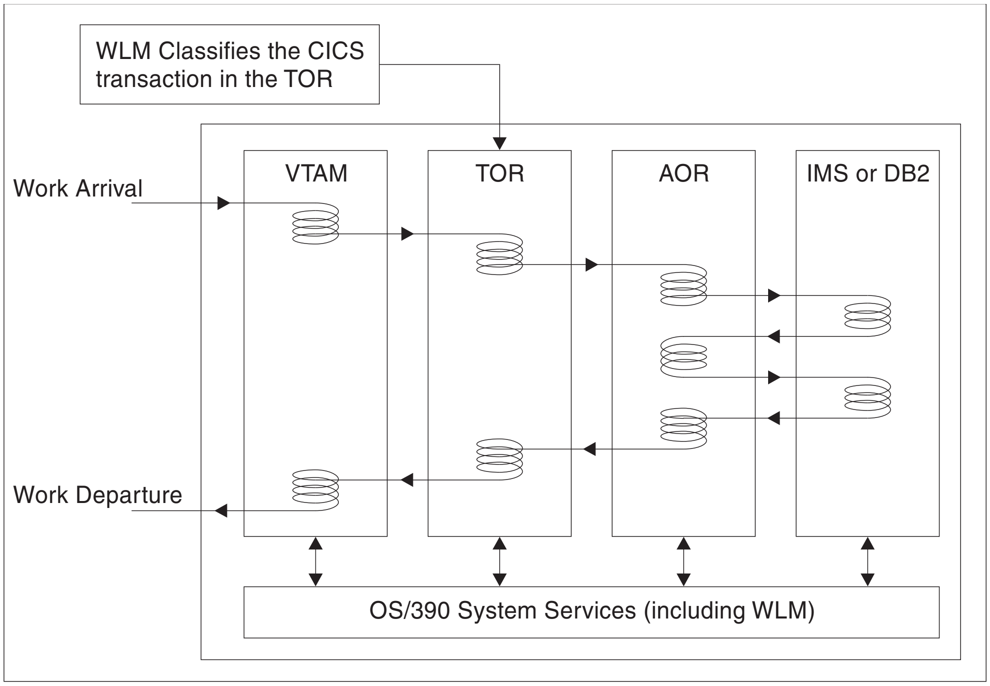


Figure 28. CICS MRO transaction workflow

z/OS Resource Measurement Facility Report Analysis has more information about understanding RMF reports.

Part 3. The CICS monitoring facility

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.

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

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.

This section covers the following topics:

Chapter 28. Collecting and processing data for CICS monitoring

You can request CICS to collect four types, or classes, of monitoring data. The collection of CICS data can be activated or deactivated and you can select when CICS data is collected. The monitoring data can then be processed by products such as the CICS Performance Analyzer or a similar application to provide information to help you analyze the performance of your system.

The DFHMNDUP utility program and the DFH$MOLS sample program are available to process CICS monitoring data that is written to system management facilities (SMF) data sets. See Sample monitoring data print program (DFH$MOLS) in Reference -> Utilities.

Class monitoring data

You can request CICS to collect four types, or classes of monitoring data: performance class data, exception class data, transaction resource data, and identity class data. You can choose which classes of monitoring data you want to be collected.

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. CICS writes at least one performance monitoring record for each transaction.

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.

This information can be used periodically to calculate the charges applicable to different tasks. If you want to set up algorithms to charge users for resources that they use, you could use this class of data collection to update the charging information in your organization's accounting programs.

You can enable performance class monitoring by coding **MNPER=ON** (together with **MN=ON**) as a system initialization parameter. Alternatively, you can use the monitoring facility transaction CEMN, or the **EXEC CICS SET MONITOR** command, to enable performance class monitoring dynamically.

CICS monitoring performance class data is collected at system-defined event monitoring points (EMPs) in the CICS code. You can choose which classes of monitoring data you want to be collected. You cannot relocate these monitoring points, but you can create additional ones, at which you can gather user-defined performance data. You define user event monitoring points by coding DFHMCT TYPE=EMP macros.

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:

- 0 - 256 counters

- 0 - 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, 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, the user EMP would have no effect.

User EMPs can use the **EXEC CICS MONITOR** command. For programming information about this command, see MONITOR in Reference -> Application development.

Additional EMPs are provided in some IBM licensed programs, such as DBCTL. From the CICS perspective, 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 entry names.

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

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

The following ideas show how you can use the CICS and user fields provided with the CICS monitoring facility:

- To time how long it takes to do a table lookup routine in an application, code an EMP with, for example, 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 one unit for each record scanned and not selected, and three 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 or operator errors. 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 other processes. It usually includes any waiting for the transaction to regain control after the requested operation has completed. The periods are counted as well as added, so you can get both the total and the average time waiting for I/O. To highlight an unusually long individual case, set a flag on in a user count, as explained earlier.

- 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.

  For example, some users have a single transaction ID so that all user input is routed through a common prologue program for security checking. In this situation, it is 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.)

For more information about event monitoring points, see User event monitoring points - DFHMCT TYPE=EMP.

Related reference:

"Performance class data: listing of data fields" on page 349
The performance class data is listed in this section in order of group name. The group name is always in field CMODNAME of the dictionary entry.

Application naming event monitoring points

You can also use application naming event monitoring points. Application naming is an enabling function that allows your application programs to invoke special CICS event monitoring points. Data collected at these CICS-generated EMPs can be used by any CICS monitoring reporting package.

For information about the APPLNAME parameter that you use to enable application naming support, see in the *CICS Resource Definition Guide*.

Examples of invoking application naming EMPs: Figure 29 shows an assembler example of how to move a CICS transaction ID to the transaction monitoring area.

```
DFHEISTG DSECT
EMPDATA1 DS    F                            Data area for DATA1 address
*
*
* Constants for DATA2 (null value) and ENTRYNAME
*
EMPDATA2 DC     F'0'
APPLNAME DC     CL8'DFHAPPL'
*
         LA     Rn,tranid  Set addr of tranid
         ST     Rn,EMPDATA1  Store tranid for EMP
         EXEC CICS MONITOR POINT(1) ENTRYNAME(APPLNAME)                 C
              DATA1(EMPDATA1) DATA2(EMPDATA2) NOHANDLE
```

Figure 29. EXEC CICS MONITOR commands for DFHAPPL EMPs (assembler)

This example shows 4 bytes of user data, typically the transaction ID, being moved using the DFHAPPL.1 EMP. The data starts at offset 0, and the data length defaults to the length specified in the application naming EMP in the MCT. In this example, CICS monitoring domain uses the default length defined in the MCT, because DATA2 is defined as a null value. For the DFHAPPL EMPs, CICS monitoring domain requires you to specify both DATA1 and DATA2.

Figure 30 on page 298 shows a COBOL example of how to move a predefined application name and a transaction identifier to the transaction monitoring area. This example uses both EMP 1 and EMP 2 of the DFHAPPL EMPs, moving 4 bytes and 8 bytes respectively, which are the default lengths defined in the MCT.

```
IDENTIFICATION DIVISION.
       PROGRAM-ID. APPLNAME.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
       77  APPLICATION-NAME-PTR        POINTER.
       77  MENU-APPLICATION-NAME       PIC X(4)    VALUE 'MENU'.
       77  PAYROLL-APPLICATION-NAME    PIC X(8)    VALUE 'PAYROLL '.
       77  DFHAPPL-NAME                PIC X(8)    VALUE 'DFHAPPL '.
       77  DFHAPPL-DATA2               PIC S9(8)   COMP   VALUE +0.
       77  BLANKS                      PIC X       VALUE SPACE.
      *
       LINKAGE SECTION.
       77  LS-APPLICATION-NAME         PIC X(8).
      *
       PROCEDURE DIVISION.
      *    Get storage for DFHAPPL data and set address
           EXEC CICS GETMAIN FLENGTH(LENGTH OF LS-APPLICATION-NAME)
               SET(APPLICATION-NAME-PTR) INITIMG(BLANKS)
           END-EXEC.
           SET ADDRESS OF LS-APPLICATION-NAME TO APPLICATION-NAME-PTR.

           MOVE PAYROLL-APPLICATION-NAME TO LS-APPLICATION-NAME.
      *    Invoke DFHAPPL EMP 2 to add the application name
           EXEC CICS MONITOR ENTRYNAME(DFHAPPL-NAME) POINT(2)
               DATA1(APPLICATION-NAME-PTR) DATA2(DFHAPPL-DATA2)
               NOHANDLE
           END-EXEC.
      *    Re-use application data area for transaction ID
           MOVE MENU-APPLICATION-NAME TO LS-APPLICATION-NAME.
      *    Invoke DFHAPPL EMP 1 to add the transaction ID
           EXEC CICS MONITOR ENTRYNAME(DFHAPPL-NAME) POINT(1)
               DATA1(APPLICATION-NAME-PTR) DATA2(DFHAPPL-DATA2)
               NOHANDLE
           END-EXEC.

           SET ADDRESS OF LS-APPLICATION-NAME TO NULL.

           EXEC CICS FREEMAIN DATAPOINTER(APPLICATION-NAME-PTR)
               NOHANDLE
           END-EXEC.

           EXEC CICS RETURN END-EXEC.
```

Figure 30. EXEC CICS MONITOR commands for DFHAPPL EMPs (COBOL)

Exception class data

Exception class monitoring data is information on CICS resource shortages that are suffered by a transaction, such as queuing for file strings, or waiting for temporary storage. This data highlights possible problems in CICS system operation, and is intended to help you identify system constraints that affect the performance of your transactions. CICS writes one exception record for each exception condition that occurs.

Exception records are produced for each of the following resource shortages:

- 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 storage in the GCDSA
- Wait for auxiliary temporary storage
- Wait for auxiliary temporary storage string
- Wait for auxiliary temporary storage buffer
- Wait for auxiliary temporary storage write buffer
- Wait for temporary storage queue
- Wait for temporary storage data set extension
- Wait for shared temporary storage
- Wait for shared temporary storage pool
- Wait for coupling facility data tables locking (request) slot
- Wait for coupling facility data tables non-locking (request) slot (With coupling facility data tables each CICS has a number of slots available for requests in the CF data table. When all available slots are in use, any further request must wait.)
- Wait for file buffer
- Wait for LSRPOOL string
- Wait for file string

Exception records are also produced when a policy threshold is exceeded.

An exception record is created each time any of the resources covered by exception class monitoring becomes constrained by system bottlenecks or a policy threshold is exceeded. The exception records are produced and written to SMF as soon as the resource shortage encountered by the transaction has been resolved or the policy threshold has been exceeded.

If performance class monitoring data is also being recorded, the performance class record for the transaction includes the total elapsed time the transaction was delayed by CICS system resource shortages, and a count of the number of exception records that have occurred for the task. The exception class records can be linked to the performance class records by either the transaction sequence number or the network unit-of-work ID.

You can enable exception class monitoring by coding MNEXC=ON (together with MN=ON) as a system initialization parameter. Alternatively you can use the monitoring facility transaction CEMN, or the CEMT or EXEC CICS SET MONITOR command, to enable exception class monitoring dynamically.

Related reference:

"Exception class data: listing of data fields" on page 393
The exception class data is listed in the order in which it appears in the exception data section of a monitoring record.

Transaction resource class data

Transaction resource class data provides additional transaction-level information about individual resources accessed by a transaction. Currently, the transaction resource class covers distributed program link, file, and temporary storage queue resources.

CICS writes one transaction resource record for each transaction that is being monitored, if the transaction accesses at least one of the resources for which monitoring data is requested.

Transaction resource data is collected when a transaction ends. You can collect information for up to a maximum of 64 distributed program links, 64 files, and 64 temporary storage queues.

The transaction resource records produced are of variable length depending on the number of resources for which data is being collected. For only one distributed program link, the record length is 436 bytes plus 32 bytes for the program data, making a total of 468 bytes. Each additional resource extends the length of the record. Each file adds 112 bytes and each temporary storage queue adds another 120 bytes. For example, if a transaction accessed 1 distributed program link, 5 files, and 2 temporary storage queues, the total length would be 436 bytes + 32 bytes + 560 bytes + 240 bytes = 1268 bytes.

Performance class data provides information about distributed program link, file, and temporary storage queue resource accesses, but this information in the performance record is given only in total, for all distributed program links, files, and temporary storage queues. Transaction resource data breaks this information down by individual distributed program link name, file name, and temporary storage queue name. The distributed program link information is held in the DFHPROG performance data group, the file information is in the DFHFILE performance data group, and the temporary storage queue information is held in the DFHTEMP performance data group.

You can enable transaction resource class monitoring at startup by coding MNRES=ON (together with MN=ON) as a system initialization parameter. Alternatively, you can use the monitoring facility transaction CEMN or the CEMT or EXEC CICS SET MONITOR command to enable transaction resource monitoring dynamically.

The maximum number of distributed program links, files, and temporary storage queues monitored for each transaction is specified by the DPL, FILE, and TSQUEUE parameters on the DFHMCT TYPE=INITIAL macro. The default is DPL=0 for distributed program links, FILE=8 for files, and TSQUEUE=8 for temporary storage queues. If transaction resource class monitoring is enabled (MNRES=ON), these defaults apply if you have not specified those options in the MCT. If the default values are insufficient, you must assemble an MCT that specifies a higher number.

If you do *not* want to collect transaction resource data for either files or temporary storage queues, but you do want to collect transaction resource data for the other resource, you must assemble an MCT that specifies FILE=0 or TSQUEUE=0 to stop transaction resource data being collected for the appropriate resource.

Transaction resource class data for a file or temporary storage queue is collected and recorded only for local resources, not for remote resources. When an application accesses a remote file or temporary storage queue, a transaction resource record is produced in the CICS region where the resource is defined locally, but no record is produced in the application-owning region.

Related reference:

"Transaction resource class data: Listing of data fields" on page 398
The transaction resource class data is listed in the order in which it appears in the

transaction resource data section of a monitoring record.

Identity class data

Identity class data provides enhanced audit information by capturing identity propagation data (an X.500 distinguished name and associated realm) from a client system across a network for eligible transactions.

Identity propagation depends on the z/OS Identity Propagation function that is provided in z/OS, Version 1 Release 11. An identity class data record is written by CICS as an SMF 110 subtype 1 record, which is created during transaction detach processing for each transaction that has identity propagation data.

You can enable identity class monitoring by coding MNIDN=ON, with MN=ON, as a system initialization parameter. Alternatively, you can use the monitoring facility transaction CEMN or the **EXEC CICS SET MONITOR** command to enable identity class monitoring dynamically.

Identity data is constructed using fields that are written only if the data is available, in a similar way to those fields used in the RACF SMF records. Unlike other monitoring SMF 110 records, these records are not compressed. The identity records are buffered (one or more identity records are constructed into a single SMF 110 record) to minimize the number of SMF writes. Any unwritten identity data records remaining in the output buffer are recorded either when the monitoring identity class is set to inactive or when CICS shuts down normally.

Related reference:

"Identity class data: Listing of data fields" on page 405
The identity class data is listed in the order in which it appears in the identity class data section of a monitoring record.

How CICS monitoring data is passed to SMF

The various CICS monitoring class records are written to SMF in different ways.

Performance data records are written to a performance record buffer, which is defined and controlled by CICS, as the records are produced. The performance records are passed to SMF for processing when the buffer is full, when the performance class of monitoring is switched off, and when CICS itself quiesces. When monitoring itself is switched off or when there is an immediate shutdown of CICS, the performance records are not written to SMF and the data is lost. When you switch off monitoring, you can use the NOPERF option on the SET MONITOR command to flush the buffers that contain recorded data for completed tasks and avoid losing the data.

Transaction resource class data records are written to a transaction resource record buffer, which is defined and controlled by CICS, as the records are produced. The transaction resource records are passed to SMF for processing when the buffer is full; when the transaction resource class of monitoring is switched off; and when CICS itself quiesces. When monitoring itself is deactivated or when there is an immediate shutdown of CICS, the transaction resource records are not written to SMF and the data is lost.

Exception records are passed directly to SMF when the exception condition completes. Each exception record describes one exception condition. You can link performance records with their associated exception records by matching the value of the TRANNUM field in each type of record; each contains the same transaction number.

Identity class records are constructed using fields that are written only if the data is available, in a similar way to those fields used in the RACF SMF records. Unlike other monitoring SMF 110 records, these records are not compressed. The identity records are buffered (one or more identity records are constructed into a single SMF 110 record) to minimize the number of SMF writes. Any unwritten identity data records remaining in the output buffer are recorded either when the monitoring identity class is set to inactive or when CICS shuts down normally.

The z/OS workload manager (WLM) and CICS monitoring facility (CMF)

Managing and monitoring your CICS workload is important to help you to achieve performance goals.

Workload management is the process of defining performance goals for the items of work in a system (such as a z/OS sysplex or a CICSplex). By using a workload manager (such as z/OS Workload Manager) to adjust resource allocations or work routing in the system in order to meet those performance goals. The z/OS workload manager (WLM) provides transaction activity reporting by service class, report class, or both, based on transaction response time information.

For more information, see “The z/OS Workload Manager” on page 273.

Monitoring data provides information which can be used to assess performance. You can analyze monitoring data reports to choose or adjust performance goals for the system, tackle any performance problems through tuning activities, and make decisions about your future strategy and investments. The CICS monitoring facility collects data about all transactions in the CICS region during online processing for later offline analysis.

Controlling CICS monitoring

You can switch CICS monitoring on or off, and select the classes of monitoring data you want to be collected, either dynamically or at CICS initialization.

About this task

When you are starting CICS, 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**, **MNRES**, **MNIDN** and **MNEXC** system initialization parameters. You can request the collection of any combination of performance class data, transaction resource class data, and exception class data. You can change the class settings whether the CICS monitoring facility is on or off. For details of all the system initialization parameters that control monitoring activities, see the *CICS 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. You can also change other settings, such as whether or not data compression is carried out for monitoring records, and the interval at which CICS produces performance class records for long-running tasks. There are two ways of controlling the monitoring facility dynamically:

1. You can use the CICS Explorer Regions view by selecting **Operations** > **Regions** from the CICS Explorer main menu.
2. You can use the **EXEC CICS INQUIRE MONITOR** and **SET MONITOR** commands. See the *CICS System Programming Reference* for information about these commands.
3. You can use the CICS monitoring facility transaction CEMN. CEMN is described in *CICS Supplied Transactions*.

If you activate a class of monitoring data while CICS is running, the data for that class becomes available only for transactions that are started after that point in time. You cannot add to 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.

If you deactivate a class of monitoring data while CICS is running, or make other changes to the settings for the monitoring facility, this affects the monitoring data which is recorded for transactions that are running at the time you make the change. Data for these transactions might be incomplete or not recorded at all, depending on the class of monitoring data involved. The documentation for the monitoring control methods listed here explains the impact of your dynamic changes.

Related information:

Specifying CICS system initialization parameters

CEMN transaction

CEMT INQUIRE MONITOR

CEMT SET MONITOR

EXEC CICS INQUIRE MONITOR

EXEC CICS SET MONITOR

Processing CICS monitoring facility output

You can process output from the CICS monitoring facility using products such as CICS Performance Analyzer and Tivoli Decision Support, or your own application program, or the supplied sample program DFH$MOLS.

CICS Performance Analyzer for z/OS
CICS Performance Analyzer (CICS PA) is a reporting tool that provides information on the performance of your CICS systems and applications. CICS Performance Analyzer is summarized in this information. For the most up to date information about CICS Performance Analyzer, see the CICS Performance Analyzer documentation.

Tivoli Decision Support for z/OS
Tivoli Decision Support for z/OS is a reporting system which uses DB2 to analyze, store and present utilization and throughput data from many sources. The Tivoli Decision Support for z/OS CICS performance feature provides reports based on data from the CICS monitoring facility and CICS statistics. For a summary of this tool, see "Tivoli Decision Support for z/OS" on page 36.

The CICS-supplied sample program DFH$MOLS
CICS provides a sample program, DFH$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.

Comments within the program can help you if you want to do your own processing of CICS monitoring facility output. See Sample monitoring data print program (DFH$MOLS) in Reference -> Utilities for more information on the DFH$MOLS program.

Your own program
You might want to write your own application program to report and analyze the data in the monitoring records.

Remember that if you have activated data compression for your SMF 110 monitoring records, the data sections of the records need to be expanded using the z/OS Data Compression and Expansion Services before they can be processed. DFH$MOLS can do this. If you are using a reporting tool, you need to ensure that it supports expansion of the data sections. "Data compression for monitoring records" on page 305 has more information on data compression.

Related information:

"CICS Performance Analyzer for z/OS (CICS PA)" on page 27
CICS Performance Analyzer (CICS PA) is a reporting tool that provides information on the performance of your CICS systems and applications, and helps you tune, manage, and plan your CICS systems effectively.

"Performance measuring with Tivoli Decision Support for z/OS" on page 37
Tivoli Decision Support for z/OS is a reporting system which uses DB2. You can use it to process utilization and throughput statistics written to log data sets by computer systems. You can use it to analyze and store the data into DB2, and present it in a variety of forms.

DFH$MOLS, sample monitoring data print program

The monitoring control table (MCT)

Use the monitoring control table (MCT) to control the nature and extent of the monitoring that you require.

- To specify the type of resource for which you want to collect transaction resource monitoring data.
- To deactivate data compression for monitoring records because data compression is the default.
- To enable application naming support, which makes available the CICS-generated DFHAPPL EMPs to your application programs.
- To specify whether you want additional monitoring performance data to be collected for the resource managers used by your transaction.
- To notify 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 notify CICS that you want certain system-defined performance data not to be recorded during a particular CICS run.

Full details of the MCT are provided in in the *CICS Resource Definition Guide*.

Four sample monitoring control tables are also provided in CICSTS53.CICS.SDFHSAMP:

- DFHMCTT$, for terminal-owning regions (TORs)
- DFHMCTA$, for application-owning regions (AORs)
- DFHMCTD$, for application-owning regions (AORs) with DBCTL
- DFHMCTF$, for file-owning regions (FORs)

These samples show you how to use the EXCLUDE and INCLUDE operands to determine the data that is included in the performance class record.

DFHMCT TYPE=INITIAL

You use the TYPE=INITIAL macro to indicate whether you want application naming support, data compression, additional performance class monitoring for the resource managers used by your transactions, and the transaction monitoring resource limit values.

For information about the APPLNAME, COMPRESS, RMI, DPL, FILE, and TSQUEUE parameters that control these facilities, see the *CICS Resource Definition Guide*.

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 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.

Related concepts:

“Data compression for monitoring records”
CICS performs data compression, by default, on the SMF 110 monitoring records produced by the CICS monitoring facility (CMF). Data compression can provide a significant reduction in the volume of data written to SMF. The records are compressed and expanded using standard z/OS services.

Data compression for monitoring records

CICS performs data compression, by default, on the SMF 110 monitoring records produced by the CICS monitoring facility (CMF). Data compression can provide a significant reduction in the volume of data written to SMF. The records are compressed and expanded using standard z/OS services.

Control data compression for monitoring records by specifying the COMPRESS option in your Monitoring Control Table (MCT), using the DFHMCT TYPE=INITIAL macro. COMPRESS=YES is the default for this option, meaning that data compression is used. If you specify the system initialization parameter

MCT=NO, the default MCT built by CICS specifies COMPRESS=YES. If you do not want to compress monitoring records, you must specify COMPRESS=NO in your MCT.

You can inquire on and change the data compression option dynamically using the monitoring facility transaction CEMN, or the equivalent EXEC CICS commands. However, when CICS is restarted the data compression option reverts to the COMPRESS value in your MCT, if you use a monitoring control table and specify the monitoring control table suffix on the MCT system initialization parameter.

When data compression is active, CICS uses the standard z/OS Data Compression and Expansion Services (CSRCESRV) to compress the CICS data section of each monitoring record before writing it to SMF. The SMF header and SMF product section of records are not compressed. This process can provide a very considerable reduction in the volume of data written to SMF, and a corresponding reduction in I/O and CPU usage for the SMF address space. If you normally exclude monitoring data fields to reduce data volume, you might find that using data compression removes the need for exclusion and enables you to collect complete monitoring data.

The collected monitoring data can include a mix of compressed records and records that have not been compressed. Records might not be compressed because of the following situations:

- Depending on the data pattern of the record, compressing the data section might possibly result in a larger record. If this situation occurs, CICS does not compress the record.
- Data compression might fail because of a problem involving the z/OS Data Compression and Expansion Services.
- Data compression might be switched off dynamically using the CEMN transaction or EXEC CICS SET MONITOR command.

When CICS SMF 110 monitoring records have been compressed, they must be identified and expanded using the z/OS Data Compression and Expansion Services, before they can be processed by SMF 110 reporting tools.

- The CICS-supplied monitoring sample program DFH$MOLS supports the expansion of compressed CICS SMF 110 monitoring records. DFH$MOLS automatically identifies any compressed monitoring records in the input, and uses the z/OS data expansion service to expand them before working with them. If you specify the EXPAND control statement, DFH$MOLS copies the compressed monitoring records to an output data set in their expanded format, with the records that were never compressed. See Sample monitoring data print program in the *CICS Operations and Utilities Guide* for further information on the DFH$MOLS program.
- If you use an SMF 110 reporting tool supplied by IBM or by another vendor, and you want to use data compression, make sure that the product can identify compressed CICS SMF 110 monitoring records and can expand the data section using the z/OS Data Compression and Expansion Services, so that the monitoring records can be processed correctly. If the reporting tool cannot work with records in this way, you might use DFH$MOLS with the EXPAND control statement to produce an output data set containing the SMF 110 monitoring records in their expanded format, for the tool to work with.

A reporting tool that is using the z/OS Data Compression and Expansion Services needs this information:

- The field SMFMNCRL in the SMF product section of the record identifies where data compression has been used for a monitoring record and gives the compressed length of the CICS data section. A zero value for this field means that data compression was not performed on the record.
- The maximum length of the CICS data section of an SMF 110 monitoring record, when expanded, is 32598 bytes.

For detailed information about the z/OS Data Compression and Expansion Services (CSRCESRV), see the *z/OS MVS Assembler Services Guide*, and the *z/OS MVS Assembler Services Reference ABE-HSP*.

Data compression applies only to SMF 110 records written by CICS monitoring, with subtype X'0001' in the record subtype field in the SMF header. It does not apply to the other types of SMF 110 records created by CICS; that is, records written by CICS journaling, CICS statistics, the TS data sharing server, the coupling facility data table (CFDT) server, and the named counter sequence number server.

Related concepts:

"The monitoring control table (MCT)" on page 304
Use the monitoring control table (MCT) to control the nature and extent of the monitoring that you require.

Related information:

Sample monitoring data print program (DFH$MOLS)

SET MONITOR

CEMN transaction

Measuring the resource usage of applications deployed on platforms

For CICS applications that are deployed on platforms, you can measure resource usage by using application context data. You can measure an application as a whole or measure and compare specific application operations.

Before you begin

To generate application context data, the application must be packaged and deployed on a platform, and the application must have a declared set of application entry points. An application entry point defines how other applications and users enter a CICS application. For more information on application entry points, see Application entry points in Product overview.

About this task

CICS adds an application context to each task at the point the application is entered. Application context data flows between tasks and across IPIC and MRO connections where application components are deployed on separate regions. Application context data identifies the application, operation, application version, and the platform in which the application is running. For more information on application context, see Application context in Product overview.

When performance class monitoring is enabled, task monitoring records include the application context values of the initial application that a task is associated with.

The current application context data is available in the task association data, and is also recorded as part of the performance class data, which provides detailed transaction-level information.

Procedure

1. In the CICS SM perspective of CICS Explorer, click **Operations** > **Regions** to open the Regions view.
2. Double-click each region in the platform to enable monitoring:
   a. Change the Monitoring Status field to ON.
   b. Change the Performance Monitoring Status field to PERF.

   CICS starts to collect monitoring records in the region. CICS writes at least one performance monitoring record for each transaction. Monitoring records are saved as SMF110 records for offline analysis.
3. Use an offline processing facility, such as CICS Performance Analyzer (CICS PA), to analyze the monitoring records. CICS PA has a plug-in for CICS Explorer so you can control CICS monitoring and analyze monitoring records in the same Eclipse environment.

Results

You activated monitoring across a set of CICS regions and collected SMF 110 records that include application context data. You used an offline processing facility to analyze the resource usage for an application.

What to do next

When appropriate, turn off monitoring in the CICS regions.

Chapter 29. Data fields for CICS monitoring data

Data fields for exception class data, identity class data, transaction resource class data, and system-defined performance class data can be produced by CICS monitoring.

Each of the data fields is presented as a field description, followed by an explanation of the contents. Here's an example of a field description:

```
001 (TYPE-C, 'TRAN', 4 BYTES)
```

The field description includes four elements: a field identifier, a data type, an informal name, and the field length. In the dictionary data section of a performance class record, these items of information are shown, along with some others, in the dictionary entry relating to the field. (Exception class data is not defined in the dictionary record.) Table 27 describes the elements of the field description and shows the corresponding element of the dictionary entry.

Table 27. Format of the descriptions of the data fields

Element	**Example**	**Description**	**Dictionary entry element**
Field identifier	`001`	A number which uniquely identifies the field within its group. The field identifier can be used in the monitoring control table (MCT) to exclude or include the field when the data is collected.	CMODIDNT
Data type	`TYPE-C`	A single letter code describing the type of data in this field. There are five data types: **A** A 32-bit count, a 64-bit count, or a 64-bit string. **C** A byte string. **P** A packed decimal value. **S** A clock. "Clocks and time stamps" on page 335 explains the components of a clock. **T** A time stamp, which is an 8-byte copy of the output of a local store clock (STCK) instruction.	CMODTYPE
Informal name	'TRAN'	A descriptive name for the field. If the monitoring output is processed into a report, this name can be used to label the field.	CMODHEAD
Length of field	4 BYTES	Some types of data always have the same field length, and others vary: **A** Field length is either 4 bytes or 8 bytes. **C** Field length varies. **P** Field length is 4 bytes (there is only one type P field). **S** Field length is always 12 bytes. **T** Field length is always 8 bytes.	CMODLENG

CICS monitoring record formats

Use this information if you write your own program to analyze the monitoring data.

CICS writes several types of SMF 110 record. Each type, or subtype as it is known, can be identified using the record subtype field in the SMF header. The subtype values are as follows:

X'0000'
CICS journaling

X'0001'
CICS monitoring

X'0002'
CICS statistics

X'0003'
Shared temporary storage queue server statistics

X'0004'
Coupling facility data table server statistics

X'0005'
Named counter sequence number server statistics

The three components of a CICS monitoring record are an SMF header, an SMF product section, and a CICS data section.

SMF Header	SMF Product Section	CICS Data Section

Figure 31. Format of an SMF type 110 monitoring record

SMF header and SMF product section

The SMF header describes the system creating the output. The SMF product section identifies the subsystem to which the monitoring data relates, which, in the case of CICS monitoring (and also of CICS statistics), is the CICS region.

Both the SMF header and the SMF product section can be mapped by the DSECT MNSMFDS, which you can generate using the DFHMNSMF macro as follows:

```
MNSMFDS DFHMNSMF PREFIX=SMF
```

The label ‘MNSMFDS' is the default DSECT name, and SMF is the default PREFIX value, so you could also generate the DSECT by coding:

```
DFHMNSMF
```

The MNSMFDS DSECT has the format shown in Figure 32 on page 311.

```
*        START OF THE SMF HEADER
*
MNSMFDS  DSECT
SMFMNLEN DS    XL2            RECORD LENGTH
SMFMNSEG DS    XL2            SEGMENT DESCRIPTOR
SMFMNFLG DS    X              OPERATING SYSTEM INDICATOR (see note 1)
SMFMNRTY DC    X'6E'          RECORD 110 FOR CICS
SMFMNTME DS    XL4            TIME RECORD MOVED TO SMF
SMFMNDTE DS    XL4            DATE RECORD MOVED TO SMF
SMFMNSID DS    CL4            SYSTEM IDENTIFICATION
SMFMNSSI DS    CL4            SUBSYSTEM IDENTIFICATION
SMFMNSTY DS    XL2            RECORD SUBTYPE - MONITORING USES TYPE 1
SMFMNTRN DS    XL2            NUMBER OF TRIPLETS
         DS    XL2            RESERVED
SMFMNAPS DS    XL4            OFFSET TO PRODUCT SECTION
SMFMNLPS DS    XL2            LENGTH OF PRODUCT SECTION
SMFMNNPS DS    XL2            NUMBER OF PRODUCT SECTIONS
SMFMNASS DS    XL4            OFFSET TO DATA SECTION
SMFMNASL DS    XL2            LENGTH OF DATA SECTION
SMFMNASN DS    XL2            NUMBER OF DATA SECTIONS
*
*        THIS CONCLUDES THE SMF HEADER
*

*
*        START OF THE SMF PRODUCT SECTION
*
SMFMNRVN DS    XL2            RECORD VERSION (CICS)
SMFMNPRN DS    CL8            PRODUCT NAME (GENERIC APPLID)
SMFMNSPN DS    CL8            PRODUCT NAME (SPECIFIC APPLID)
SMFMNMFL DS    XL2            RECORD MAINTENANCE INDICATOR
         DS    XL2            RESERVED
SMFMNCL  DS    XL2            CLASS OF DATA
*                                         1 = DICTIONARY
*                                         3 = PERFORMANCE
*                                         4 = EXCEPTION
*                                         5 = TRANSACTION RESOURCE
*                                         6 = IDENTITY
SMFMNDCA DS    XL4            OFFSET TO CICS FIELD CONNECTORS
SMFMNDCL DS    XL2            LENGTH OF EACH CICS FIELD CONNECTOR
SMFMNDCN DS    XL2            NUMBER OF CICS FIELD CONNECTORS
SMFMNDRA DS    XL4            OFFSET TO FIRST CICS DATA RECORD
SMFMNDRL DS    XL2            LENGTH OF EACH CICS DATA RECORD
SMFMNDRN DS    XL2            NUMBER OF CICS DATA RECORDS
*
         DS    XL18           RESERVED
SMFMNCRL DS    XL2            COMPRESSED RECORD LENGTH (see note 7)
SMFMNTAD DS    XL4            LOCAL TOD CLOCK ADJUSTMENT VALUE
SMFMNLSO DS    XL8            LEAP SECOND OFFSET TOD FORMAT
SMFMNDTO DS    XL8            LOCAL TIME/DATE OFFSET
         DS    XL1            RESERVED
SMFMNOPN DS    XL1            MONITORING OPTIONSSMFMNJBN DS    CL8           JOBNAME
SMFMNRSD DS    XL4            JOB DATE
SMFMSRST DS    XL4            JOB TIME
SMFMNUIF DS    CL8            USER IDENTIFICATION
SMFMNPDN DS    CL8            OPERATING SYSTEM PRODUCT LEVEL
*
*        THIS CONCLUDES THE SMF PRODUCT SECTION
```

Figure 32. Format of the SMF header and product section for monitoring records

Note:

1. CICS sets only the subsystem-related bits of the operating system indicator flag byte in the SMF header (SMFMNFLG). SMF sets the remainder of the byte according to the operating system level and other factors.

2. Fields SMFMNDCA SMFMNDCL, and SMFMNDCN apply to performance class records only.
3. For dictionary class monitoring records, see "Dictionary data sections" on page 313, the fields SMFMNDRA, SMFMNDRL, and SMFMNDRN in the SMF product section have the following meaning:

   SMFMNDRA
   Offset to the first dictionary entry.

   SMFMNDRL
   Length of a single dictionary entry.

   SMFMNDRN
   Number of dictionary entries within the CICS data section.
4. For performance class and exception class monitoring records, the fields SMFMNDRA, SMFMNDRL, and SMFMNDRN, in the SMF product section have the following meaning:

   SMFMNDRA
   Offset to the first performance or exception class record.

   SMFMNDRL
   Length of each performance or exception class record.

   SMFMNDRN
   Number of performance class records in the data section, but for exception class records this value is always 1.
5. For transaction resource monitoring records, the fields SMFMNDRA, SMFMNDRL, and SMFMNDRN, in the SMF product section have the following meaning:

   SMFMNDRA
   Offset to the first transaction resource monitoring record.

   SMFMNDRL
   This value is always zero because the transaction resource records in the data section are variable length. The length of each record is in the halfword field MNR_LENGTH at the start of each record.

   SMFMNDRN
   Number of transaction resource monitoring records in the data section.
6. The copy book DFHSMFDS is also provided and can be used to map the SMF header and the SMF product sections of all six subtypes of SMF 110 records written by CICS journaling, CICS monitoring, CICS statistics, the TS data sharing server, the coupling facility data table (CFDT) server, and the named counter sequence number server.
7. The SMFMNCRL field indicates whether the CICS data section in this monitoring SMF 110 record contains compressed data. A zero value in this field indicates that the CICS data section in the record does not contain compressed data. A non-zero value in this field indicates that the CICS data section in the record does contain compressed data, and that the z/OS Data Compression and Expansion Services must be used to expand the data section before processing. For more information on data compression, see "Data compression for monitoring records" on page 305.

CICS data section

The CICS data section can be made up of a dictionary data section, a performance data section, an exception data section, an identity class data section or a transaction resource data section. You can identify which of these you are dealing with by looking at the value of field SMFMNCL in the SMF product section.

If data compression has been used, the z/OS Data Compression and Expansion Services must be used to expand the data section before processing. For more information on data compression, see “Data compression for monitoring records” on page 305.

Dictionary data sections

Dictionary data sections describe all the fields in the performance data records that are gathered during this CICS run.

Dictionary data sections describe all the system-provided data fields (whether you have excluded any or not), plus any user-provided data fields, which CICS takes at initialization time from the MCT entries you have coded. This means that the descriptions of the system-provided data fields never change, though the user data fields can be changed each time CICS is initialized. The contents of the dictionary data sections cannot be changed while CICS is running.

Dictionary data sections contain a variable number of 26-byte dictionary entries. Each dictionary entry provides the following information about a single performance record data field:

CMODNAME
The identifier of the group to which the field belongs.

CMODTYPE
The field type.

CMODIDNT
The field identifier.

CMODLENG
The length of the field.

CMODCONN
The connector value assigned to the field.

CMODOFST
The offset of the field.

CMODHEAD
The informal name of the field.

You can map the dictionary entries by generating a DSECT with the DFHMCTDR macro, as shown in Figure 33 on page 314.

```
DFHMCTDR TYPE=(PREFIX,CMO)
```

CMO is the default label prefix. The DSECT is as follows:

```
CMODNAME DS    CL8     + 0    NAME OF OWNER (entry name)
CMODTYPE DS    C       + 8    OBJECT TYPE
*                           'S' = STOPWATCH (CLOCK)
*                           'A' = ACCUMULATOR (COUNT)
*                           'C' = BYTE-STRING FIELD
*                           'T' = TIMESTAMP (STCK FORMAT)
*                           'P' = PACKED-DECIMAL FIELD
CMODIDNT DS    CL3     +9     ID WITHIN TYPE
*                             CLOCK-, COUNT-, OR FIELD-NO.
CMODLENG DS    H       +12    LENGTH OF OBJECT
CMODCONN DS    XL2     +14    ASSIGNED CONNECTOR
CMODOFST DS    XL2     +16    ASSIGNED OFFSET
CMODHEAD DS    CL8     +18    INFORMAL NAME
CMODNEXT EQU   *
```

Figure 33. CICS monitoring dictionary entry DSECT

Whenever the monitoring of performance class data is switched on, whether at CICS initialization or while CICS is running, a dictionary data section is written. So, if the monitoring of performance class data is switched on and off three times during a single CICS run, there are three separate, but identical, dictionary data sections for that run. The dictionary data section is passed to SMF, together with any performance data sections, when the first buffer of performance data sections for a performance class data monitoring session is output to SMF. Any offline utility should use the most recent dictionary record encountered when processing CICS monitoring records.

The format of dictionary data sections is shown in Figure 34.

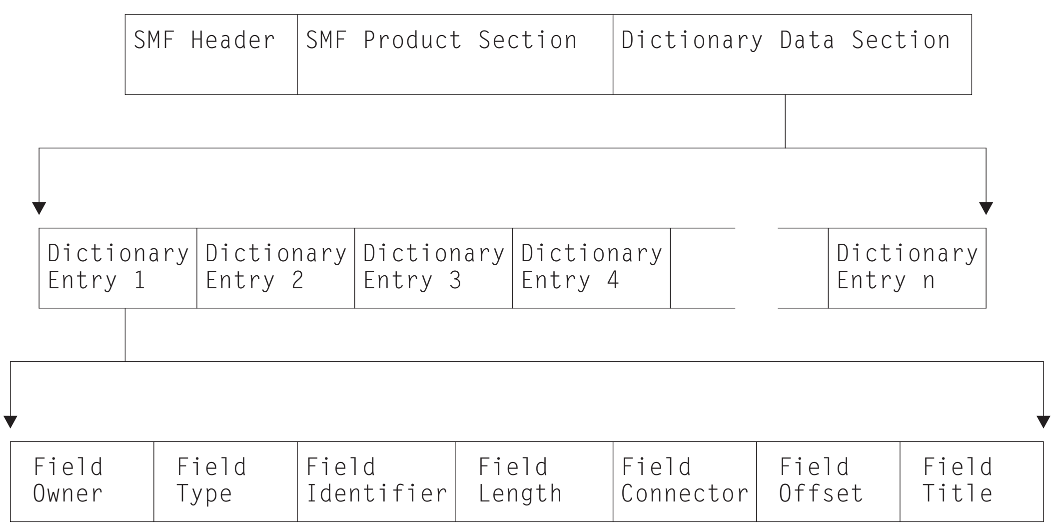


Figure 34. Format of the CICS monitoring dictionary data section

Default CICS dictionary entries

The dictionary data sections of CICS monitoring SMF type 110 records contain default CICS dictionary entries.

The field types, which start the second part of the field name, are as follows:

A Count

C Byte string

P Packed decimal number

S Clock

T Time stamp

Note: Nicknames might not be unique.

```
FIELD-NAME      SIZE  CONNECTOR OFFSET    NICKNAME

DFHTASK  C001      4  X'0001'   X'0000'   TRAN
DFHTERM  C002      4  X'0002'   X'0004'   TERM
DFHCICS  C089      8  X'0003'   X'0008'   USERID
DFHTASK  C004      4  X'0004'   X'0010'   TTYPE
DFHCICS  T005      8  X'0005'   X'0014'   START
DFHCICS  T006      8  X'0006'   X'001C'   STOP
DFHTASK  P031      4  X'0007'   X'0024'   TRANNUM
DFHTASK  A109      4  X'0008'   X'0028'   TRANPRI
DFHTASK  C166      8  X'0009'   X'002C'   TCLSNAME
DFHTERM  C111      8  X'000A'   X'0034'   LUNAME
DFHPROG  C071      8  X'000B'   X'003C'   PGMNAME
DFHTASK  C097     20  X'000C'   X'0044'   NETUOWPX
DFHTASK  C098      8  X'000D'   X'0058'   NETUOWSX
DFHCICS  C130      4  X'000E'   X'0060'   RSYSID
DFHCICS  A131      4  X'000F'   X'0064'   PERRECNT
DFHTASK  T132      8  X'0010'   X'0068'   RMUOWID
DFHCICS  C167      8  X'0011'   X'0070'   SRVCLSNM
DFHCICS  C168      8  X'0012'   X'0078'   RPTCLSNM
DFHTASK  C163      4  X'0013'   X'0080'   FCTYNAME
DFHTASK  A164      8  X'0014'   X'0084'   TRANFLAG
DFHTERM  A165      4  X'0015'   X'008C'   TERMINFO
DFHTERM  C169      4  X'0016'   X'0090'   TERMCNNM
DFHTASK  C124      4  X'0017'   X'0094'   BRDGTRAN
DFHTASK  C190     16  X'0018'   X'0098'   RRMSURID
DFHCBTS  C200     36  X'0019'   X'00A8'   PRCSNAME
DFHCBTS  C201      8  X'001A'   X'00CC'   PRCSTYPE
DFHCBTS  C202     52  X'001B'   X'00D4'   PRCSID
DFHCBTS  C203     52  X'001C'   X'0108'   ACTVTYID
DFHCBTS  C204     16  X'001D'   X'013C'   ACTVTYNM
DFHSOCK  C318     40  X'001E'   X'014C'   CLIPADDR
DFHTASK  C082     28  X'001F'   X'0174'   TRNGRPID
DFHTERM  C197      8  X'0020'   X'0190'   NETID
DFHTERM  C198      8  X'0021'   X'0198'   RLUNAME
DFHSOCK  C245      8  X'0022'   X'01A0'   TCPSRVCE
DFHSOCK  A246      4  X'0023'   X'01A8'   PORTNUM
DFHTASK  C194    128  X'0024'   X'01AC'   OTSTID
DFHSOCK  A330      4  X'0025'   X'022C'   CLIPPORT
DFHSOCK  C305      8  X'0026'   X'0230'   ISIPICNM
DFHCICS  C359      8  X'0027'   X'0238'   ONETWKID
DFHCICS  C360      8  X'0028'   X'0240'   OAPPLID
DFHCICS  T361      8  X'0029'   X'0248'   OSTART
DFHCICS  P362      4  X'002A'   X'0250'   OTRANNUM
DFHCICS  C363      4  X'002B'   X'0254'   OTRAN
DFHCICS  C364      8  X'002C'   X'0258'   OUSERID
DFHCICS  C365     64  X'002D'   X'0260'   OUSERCOR
DFHCICS  C366      8  X'002E'   X'02A0'   OTCPSVCE
DFHCICS  A367      4  X'002F'   X'02A8'   OPORTNUM
DFHCICS  C372     40  X'0030'   X'02AC'   OCLIPADR
DFHCICS  A369      4  X'0031'   X'02D4'   OCLIPORT
DFHCICS  A370      8  X'0032'   X'02D8'   OTRANFLG
DFHCICS  C371      8  X'0033'   X'02E0'   OFCTYNME
DFHWEBB  C380      8  X'0034'   X'02E8'   WBURIMNM
DFHWEBB  C381      8  X'0035'   X'02F0'   WBPIPLNM
DFHWEBB  C382      8  X'0036'   X'02F8'   WBATMSNM
DFHWEBB  C383     32  X'0037'   X'0300'   WBSVCENM
DFHWEBB  C384     64  X'0038'   X'0320'   WBSVOPNM
DFHWEBB  C385      8  X'0039'   X'0360'   WBPROGNM
DFHCICS  C373      8  X'003A'   X'0368'   PHNTWKID
DFHCICS  C374      8  X'003B'   X'0370'   PHAPPLID
DFHCICS  T375      8  X'003C'   X'0378'   PHSTART
```

Figure 35. Default CICS dictionary entries (part 1)

```
FIELD-NAME      SIZE  CONNECTOR OFFSET    NICKNAME

DFHCICS  P376      4  X'003D'   X'0380'   PHTRANNO
DFHCICS  C377      4  X'003E'   X'0384'   PHTRAN
DFHCICS  A378      4  X'003F'   X'0388'   PHCOUNT
DFHCICS  C351     64  X'0040'   X'038C'   OADID
DFHCICS  C352     64  X'0041'   X'03CC'   OADATA1
DFHCICS  C353     64  X'0042'   X'040C'   OADATA2
DFHCICS  C354     64  X'0043'   X'044C'   OADATA3
DFHSOCK  A320      4  X'0044'   X'048C'   SOCIPHER
DFHTASK  C430      4  X'0045'   X'0490'   CECMCHTP
DFHTASK  C431     16  X'0046'   X'0494'   CECMDLID
DFHTASK  A433      4  X'0047'   X'04A4'   MAXTASKS
DFHTASK  A434      4  X'0048'   X'04A8'   CURTASKS
DFHTASK  C451     64  X'0049'   X'04AC'   ACAPPLNM
DFHTASK  C452     64  X'004A'   X'04EC'   ACPLATNM
DFHTASK  A453      4  X'004B'   X'052C'   ACMAJVER
DFHTASK  A454      4  X'004C'   X'0530'   ACMINVER
DFHTASK  A455      4  X'004D'   X'0534'   ACMICVER
DFHTASK  C456     64  X'004E'   X'0538'   ACOPERNM
DFHTASK  C064      4  X'004F'   X'0578'   TASKFLAG
DFHPROG  C113      4  X'0050'   X'057C'   ABCODEO
DFHPROG  C114      4  X'0051'   X'0580'   ABCODEC
DFHCICS  C112      4  X'0052'   X'0584'   RTYPE
DFHTERM  A034      4  X'0053'   X'0588'   TCMSGIN1
DFHTERM  A083      4  X'0054'   X'058C'   TCCHRIN1
DFHTERM  A035      4  X'0055'   X'0590'   TCMSGOU1
DFHTERM  A084      4  X'0056'   X'0594'   TCCHROU1
DFHTERM  A067      4  X'0057'   X'0598'   TCMSGIN2
DFHTERM  A085      4  X'0058'   X'059C'   TCCHRIN2
DFHTERM  A068      4  X'0059'   X'05A0'   TCMSGOU2
DFHTERM  A086      4  X'005A'   X'05A4'   TCCHROU2
DFHTERM  A135      4  X'005B'   X'05A8'   TCM62IN2
DFHTERM  A137      4  X'005C'   X'05AC'   TCC62IN2
DFHTERM  A136      4  X'005D'   X'05B0'   TCM62OU2
DFHTERM  A138      4  X'005E'   X'05B4'   TCC62OU2
DFHTERM  A069      4  X'005F'   X'05B8'   TCALLOCT
DFHSTOR  A054      4  X'0060'   X'05BC'   SCUGETCT
DFHSTOR  A105      4  X'0061'   X'05C0'   SCUGETCT
DFHSTOR  A117      4  X'0062'   X'05C4'   SCCGETCT
DFHSTOR  A120      4  X'0063'   X'05C8'   SCCGETCT
DFHSTOR  A033      4  X'0064'   X'05CC'   SCUSRHWM
DFHSTOR  A106      4  X'0065'   X'05D0'   SCUSRHWM
DFHSTOR  A116      4  X'0066'   X'05D4'   SC24CHWM
DFHSTOR  A119      4  X'0067'   X'05D8'   SC31CHWM
DFHSTOR  A095      8  X'0068'   X'05DC'   SCUSRSTG
DFHSTOR  A107      8  X'0069'   X'05E4'   SCUSRSTG
DFHSTOR  A118      8  X'006A'   X'05EC'   SC24COCC
DFHSTOR  A121      8  X'006B'   X'05F4'   SC31COCC
DFHSTOR  A144      4  X'006C'   X'05FC'   SC24SGCT
DFHSTOR  A145      4  X'006D'   X'0600'   SC24GSHR
DFHSTOR  A146      4  X'006E'   X'0604'   SC24FSHR
DFHSTOR  A147      4  X'006F'   X'0608'   SC31SGCT
DFHSTOR  A148      4  X'0070'   X'060C'   SC31GSHR
DFHSTOR  A149      4  X'0071'   X'0610'   SC31FSHR
DFHSTOR  A441      4  X'0072'   X'0614'   SC64CGCT
DFHSTOR  A442      4  X'0073'   X'0618'   SC64CHWM
DFHSTOR  A443      4  X'0074'   X'061C'   SC64UGCT
DFHSTOR  A444      4  X'0075'   X'0620'   SC64UHWM
DFHSTOR  A445      4  X'0076'   X'0624'   SC64SGCT
DFHSTOR  A446      4  X'0077'   X'0628'   SC64GSHR
DFHSTOR  A447      4  X'0078'   X'062C'   SC64FSHR
DFHSTOR  A087      4  X'0079'   X'0630'   PCSTGHWM
DFHSTOR  A139      4  X'007A'   X'0634'   PC31AHWM
```

Figure 36. Default CICS dictionary entries (part 2)

FIELD-NAME		SIZE	CONNECTOR	OFFSET	NICKNAME
DFHSTOR	A108	4	X'007B'	X'0638'	PC24BHWM
DFHSTOR	A142	4	X'007C'	X'063C'	PC31CHWM
DFHSTOR	A143	4	X'007D'	X'0640'	PC24CHWM
DFHSTOR	A122	4	X'007E'	X'0644'	PC31RHWM
DFHSTOR	A162	4	X'007F'	X'0648'	PC24RHWM
DFHSTOR	A161	4	X'0080'	X'064C'	PC31SHWM
DFHSTOR	A160	4	X'0081'	X'0650'	PC24SHWM
DFHFILE	A036	4	X'0082'	X'0654'	FCGETCT
DFHFILE	A037	4	X'0083'	X'0658'	FCPUTCT
DFHFILE	A038	4	X'0084'	X'065C'	FCBRWCT
DFHFILE	A039	4	X'0085'	X'0660'	FCADDCT
DFHFILE	A040	4	X'0086'	X'0664'	FCDELCT
DFHFILE	A093	4	X'0087'	X'0668'	FCTOTCT
DFHFILE	A070	4	X'0088'	X'066C'	FCAMCT
DFHDEST	A041	4	X'0089'	X'0670'	TDGETCT
DFHDEST	A042	4	X'008A'	X'0674'	TDPUTCT
DFHDEST	A043	4	X'008B'	X'0678'	TDPURCT
DFHDEST	A091	4	X'008C'	X'067C'	TDTOTCT
DFHTEMP	A044	4	X'008D'	X'0680'	TSGETCT
DFHTEMP	A046	4	X'008E'	X'0684'	TSPUTACT
DFHTEMP	A047	4	X'008F'	X'0688'	TSPUTMCT
DFHTEMP	A460	4	X'0090'	X'068C'	TSGETSCT
DFHTEMP	A461	4	X'0091'	X'0690'	TSPUTSCT
DFHTEMP	A092	4	X'0092'	X'0694'	TSTOTCT
DFHMAPP	A050	4	X'0093'	X'0698'	BMSMAPCT
DFHMAPP	A051	4	X'0094'	X'069C'	BMSINCT
DFHMAPP	A052	4	X'0095'	X'06A0'	BMSOUTCT
DFHMAPP	A090	4	X'0096'	X'06A4'	BMSTOTCT
DFHPROG	A055	4	X'0097'	X'06A8'	PCLINKCT
DFHPROG	A056	4	X'0098'	X'06AC'	PCXCTLCT
DFHPROG	A057	4	X'0099'	X'06B0'	PCLOADCT
DFHPROG	A072	4	X'009A'	X'06B4'	PCLURMCT
DFHPROG	A073	4	X'009B'	X'06B8'	PCDPLCT
DFHPROG	A286	4	X'009C'	X'06BC'	PCDLCSDL
DFHPROG	A287	4	X'009D'	X'06C0'	PCDLCRDL
DFHPROG	A306	4	X'009E'	X'06C4'	PCLNKCCT
DFHPROG	A307	4	X'009F'	X'06C8'	PCXCLCCT
DFHPROG	A308	4	X'00A0'	X'06CC'	PCDPLCCT
DFHPROG	A309	4	X'00A1'	X'06D0'	PCRTNCCT
DFHPROG	A310	4	X'00A2'	X'06D4'	PCRTNCDL
DFHJOUR	A058	4	X'00A3'	X'06D8'	JNLWRTCT
DFHJOUR	A172	4	X'00A4'	X'06DC'	LOGWRTCT
DFHTASK	A059	4	X'00A5'	X'06E0'	ICPUINCT
DFHTASK	A066	4	X'00A6'	X'06E4'	ICTOTCT
DFHTASK	A065	4	X'00A7'	X'06E8'	ICSTACCT
DFHTASK	A345	4	X'00A8'	X'06EC'	ICSTACDL
DFHTASK	A346	4	X'00A9'	X'06F0'	ICSTRCCT
DFHTASK	A347	4	X'00AA'	X'06F4'	ICSTRCDL
DFHSYNC	A060	4	X'00AB'	X'06F8'	SPSYNCCT
DFHCICS	A025	4	X'00AC'	X'06FC'	CFCAPICT
DFHFEPI	A150	4	X'00AD'	X'0700'	SZALLOCT
DFHFEPI	A151	4	X'00AE'	X'0704'	SZRCVCT
DFHFEPI	A152	4	X'00AF'	X'0708'	SZSENDCT
DFHFEPI	A153	4	X'00B0'	X'070C'	SZSTRTCT
DFHFEPI	A154	4	X'00B1'	X'0710'	SZCHROUT
DFHFEPI	A155	4	X'00B2'	X'0714'	SZCHRIN
DFHFEPI	A157	4	X'00B3'	X'0718'	SZALLCTO
DFHFEPI	A158	4	X'00B4'	X'071C'	SZRCVTO
DFHFEPI	A159	4	X'00B5'	X'0720'	SZTOTCT
DFHCBTS	A205	4	X'00B6'	X'0724'	BARSYNCT
DFHCBTS	A206	4	X'00B7'	X'0728'	BARASYCT
DFHCBTS	A207	4	X'00B8'	X'072C'	BALKPACT
DFHCBTS	A208	4	X'00B9'	X'0730'	BADPROCT
DFHCBTS	A209	4	X'00BA'	X'0734'	BADACTCT
DFHCBTS	A210	4	X'00BB'	X'0738'	BARSPACT

Figure 37. Default CICS dictionary entries (part 3)

```
FIELD-NAME      SIZE  CONNECTOR OFFSET    NICKNAME

DFHCBTS  A211      4  X'00BC'   X'073C'   BASUPACT
DFHCBTS  A212      4  X'00BD'   X'0740'   BARMPACT
DFHCBTS  A213      4  X'00BE'   X'0744'   BADCPACT
DFHCBTS  A214      4  X'00BF'   X'0748'   BAACQPCT
DFHCBTS  A215      4  X'00C0'   X'074C'   BATOTPCT
DFHCBTS  A216      4  X'00C1'   X'0750'   BAPRDCCT
DFHCBTS  A217      4  X'00C2'   X'0754'   BAACDCCT
DFHCBTS  A218      4  X'00C3'   X'0758'   BATOTCCT
DFHCBTS  A219      4  X'00C4'   X'075C'   BARATECT
DFHCBTS  A220      4  X'00C5'   X'0760'   BADFIECT
DFHCBTS  A221      4  X'00C6'   X'0764'   BATIAECT
DFHCBTS  A222      4  X'00C7'   X'0768'   BATOTECT
DFHWEBB  A231      4  X'00C8'   X'076C'   WBRCVCT
DFHWEBB  A232      4  X'00C9'   X'0770'   WBCHRIN
DFHWEBB  A233      4  X'00CA'   X'0774'   WBSENDCT
DFHWEBB  A234      4  X'00CB'   X'0778'   WBCHROUT
DFHWEBB  A235      4  X'00CC'   X'077C'   WBTOTCT
DFHWEBB  A236      4  X'00CD'   X'0780'   WBREPRCT
DFHWEBB  A237      4  X'00CE'   X'0784'   WBREPWCT
DFHWEBB  A238      4  X'00CF'   X'0788'   WBEXTRCT
DFHWEBB  A239      4  X'00D0'   X'078C'   WBBRWCT
DFHWEBB  A224      4  X'00D1'   X'0790'   WBREADCT
DFHWEBB  A225      4  X'00D2'   X'0794'   WBWRITCT
DFHDOCH  A226      4  X'00D3'   X'0798'   DHCRECT
DFHDOCH  A227      4  X'00D4'   X'079C'   DHINSCT
DFHDOCH  A228      4  X'00D5'   X'07A0'   DHSETCT
DFHDOCH  A229      4  X'00D6'   X'07A4'   DHRETCT
DFHDOCH  A223      4  X'00D7'   X'07A8'   DHDELCT
DFHDOCH  A230      4  X'00D8'   X'07AC'   DHTOTCT
DFHDOCH  A240      4  X'00D9'   X'07B0'   DHTOTDCL
DFHSOCK  A242      4  X'00DA'   X'07B4'   SOBYENCT
DFHSOCK  A243      4  X'00DB'   X'07B8'   SOBYDECT
DFHSOCK  A289      4  X'00DC'   X'07BC'   SOEXTRCT
DFHSOCK  A290      4  X'00DD'   X'07C0'   SOCNPSCT
DFHSOCK  A291      4  X'00DE'   X'07C4'   SOCPSCT
DFHSOCK  A292      4  X'00DF'   X'07C8'   SONPSHWM
DFHSOCK  A293      4  X'00E0'   X'07CC'   SOPSHWM
DFHSOCK  A294      4  X'00E1'   X'07D0'   SORCVCT
DFHSOCK  A295      4  X'00E2'   X'07D4'   SOCHRIN
DFHSOCK  A296      4  X'00E3'   X'07D8'   SOSENDCT
DFHSOCK  A297      4  X'00E4'   X'07DC'   SOCHROUT
DFHSOCK  A298      4  X'00E5'   X'07E0'   SOTOTCT
DFHSOCK  A301      4  X'00E6'   X'07E4'   SOMSGIN1
DFHSOCK  A302      4  X'00E7'   X'07E8'   SOCHRIN1
DFHSOCK  A303      4  X'00E8'   X'07EC'   SOMSGOU1
DFHSOCK  A304      4  X'00E9'   X'07F0'   SOCHROU1
DFHDATA  A179      4  X'00EA'   X'07F4'   IMSREQCT
DFHDATA  A180      4  X'00EB'   X'07F8'   DB2REQCT
DFHDATA  A395      4  X'00EC'   X'07FC'   WMQREQCT
DFHTASK  A251      4  X'00ED'   X'0800'   TCBATTCT
DFHTASK  A252      4  X'00EE'   X'0804'   DSTCBHWM
DFHWEBB  A331      4  X'00EF'   X'0808'   WBREDOCT
DFHWEBB  A332      4  X'00F0'   X'080C'   WBWRTOCT
DFHWEBB  A333      4  X'00F1'   X'0810'   WBRCVIN1
DFHWEBB  A334      4  X'00F2'   X'0814'   WBCHRIN1
DFHWEBB  A335      4  X'00F3'   X'0818'   WBSNDOU1
DFHWEBB  A336      4  X'00F4'   X'081C'   WBCHROU1
DFHWEBB  A337      4  X'00F5'   X'0820'   WBPARSCT
DFHWEBB  A338      4  X'00F6'   X'0824'   WBBRWOCT
DFHWEBB  A340      4  X'00F7'   X'0828'   WBIWBSCT
DFHWEBB  A341      4  X'00F8'   X'082C'   WBREPRDL
DFHWEBB  A342      4  X'00F9'   X'0830'   WBREPWDL
```

Figure 38. Default CICS dictionary entries (part 4)

FIELD-NAME		SIZE	CONNECTOR	OFFSET	NICKNAME
DFHCHNL	A321	4	X'00FA'	X'0834'	PGTOTCCT
DFHCHNL	A322	4	X'00FB'	X'0838'	PGBRWCCT
DFHCHNL	A323	4	X'00FC'	X'083C'	PGGETCCT
DFHCHNL	A324	4	X'00FD'	X'0840'	PGPUTCCT
DFHCHNL	A325	4	X'00FE'	X'0844'	PGMOVCCT
DFHCHNL	A326	4	X'00FF'	X'0848'	PGGETCDL
DFHCHNL	A327	4	X'0100'	X'084C'	PGPUTCDL
DFHCHNL	A328	4	X'0101'	X'0850'	PGCRECCT
DFHCHNL	A329	4	X'0102'	X'0854'	PGCSTHWM
DFHSOCK	A288	4	X'0103'	X'0858'	ISALLOCT
DFHCICS	A402	4	X'0104'	X'085C'	EICTOTCT
DFHCICS	A415	4	X'0105'	X'0860'	ECSIGECT
DFHCICS	A416	4	X'0106'	X'0864'	ECEFOPCT
DFHCICS	A417	4	X'0107'	X'0868'	ECEVNTCT
DFHCICS	A418	4	X'0108'	X'086C'	ECSEVCCT
DFHCICS	A405	4	X'0109'	X'0870'	TIASKTCT
DFHCICS	A406	4	X'010A'	X'0874'	TITOTCT
DFHCICS	A408	4	X'010B'	X'0878'	BFDGSTCT
DFHCICS	A409	4	X'010C'	X'087C'	BFTOTCT
DFHWEBB	A412	4	X'010D'	X'0880'	MLXSSTDL
DFHWEBB	A413	4	X'010E'	X'0884'	MLXMLTCT
DFHWEBB	A420	4	X'010F'	X'0888'	WSACBLCT
DFHWEBB	A421	4	X'0110'	X'088C'	WSACGTCT
DFHWEBB	A422	4	X'0111'	X'0890'	WSAEPCCT
DFHWEBB	A423	4	X'0112'	X'0894'	WSATOTCT
DFHWEBB	A386	4	X'0113'	X'0898'	WBSFCRCT
DFHWEBB	A387	4	X'0114'	X'089C'	WBSFTOCT
DFHWEBB	A388	4	X'0115'	X'08A0'	WBISSFCT
DFHWEBB	A390	4	X'0116'	X'08A4'	WBSREQBL
DFHWEBB	A392	4	X'0117'	X'08A8'	WBSRSPBL
DFHWEBB	A424	4	X'0118'	X'08AC'	WBJSNRQL
DFHWEBB	A425	4	X'0119'	X'08B0'	WBJSNRPL
DFHCICS	A449	4	X'011A'	X'08B4'	MPPRTXCD
DFHCICS	A464	4	X'011B'	X'08B8'	NCGETCT
DFHTASK	S007	12	X'011C'	X'08BC'	USRDISPT
DFHTASK	S008	12	X'011D'	X'08C8'	USRCPUT
DFHTASK	S436	12	X'011E'	X'08D4'	CPUTONCP
DFHTASK	S437	12	X'011F'	X'08E0'	OFFLCPUT
DFHTASK	S014	12	X'0120'	X'08EC'	SUSPTIME
DFHTASK	S102	12	X'0121'	X'08F8'	DISPWTT
DFHTASK	S255	12	X'0122'	X'0904'	QRDISPT
DFHTASK	S256	12	X'0123'	X'0910'	QRCPUT
DFHTASK	S257	12	X'0124'	X'091C'	MSDISPT
DFHTASK	S258	12	X'0125'	X'0928'	MSCPUT
DFHTASK	S269	12	X'0126'	X'0934'	RODISPT
DFHTASK	S270	12	X'0127'	X'0940'	ROCPUT
DFHTASK	S262	12	X'0128'	X'094C'	KY8DISPT
DFHTASK	S263	12	X'0129'	X'0958'	KY8CPUT
DFHTASK	S264	12	X'012A'	X'0964'	KY9DISPT
DFHTASK	S265	12	X'012B'	X'0970'	KY9CPUT
DFHTASK	S259	12	X'012C'	X'097C'	L8CPUT
DFHTASK	S266	12	X'012D'	X'0988'	L9CPUT
DFHTASK	S261	12	X'012E'	X'0994'	S8CPUT
DFHTASK	S271	12	X'012F'	X'09A0'	X8CPUT
DFHTASK	S272	12	X'0130'	X'09AC'	X9CPUT
DFHTASK	S400	12	X'0131'	X'09B8'	T8CPUT
DFHTASK	S249	12	X'0132'	X'09C4'	QRMODDLY
DFHTASK	S250	12	X'0133'	X'09D0'	MAXOTDLY
DFHTASK	S282	12	X'0134'	X'09DC'	MAXXTDLY
DFHTASK	S281	12	X'0135'	X'09E8'	MAXSTDLY
DFHTASK	S283	12	X'0136'	X'09F4'	MAXTTDLY
DFHTASK	S268	12	X'0137'	X'0A00'	DSTCBMWT
DFHTASK	S247	12	X'0138'	X'0A0C'	DSCHMDLY
DFHCICS	S103	12	X'0139'	X'0A18'	EXWTTIME

Figure 39. Default CICS dictionary entries (part 5)

```
FIELD-NAME       SIZE  CONNECTOR OFFSET    NICKNAME

DFHTERM  S009     12   X'013A'   X'0A24'   TCIOWTT
DFHFILE  S063     12   X'013B'   X'0A30'   FCIOWTT
DFHFILE  S426     12   X'013C'   X'0A3C'   FCXCWTT
DFHFILE  S427     12   X'013D'   X'0A48'   FCVSWTT
DFHJOUR  S010     12   X'013E'   X'0A54'   JCIOWTT
DFHTEMP  S011     12   X'013F'   X'0A60'   TSIOWTT
DFHTERM  S100     12   X'0140'   X'0A6C'   IRIOWTT
DFHDEST  S101     12   X'0141'   X'0A78'   TDIOWTT
DFHPROG  S115     12   X'0142'   X'0A84'   PCLOADTM
DFHTASK  S125     12   X'0143'   X'0A90'   DSPDELAY
DFHTASK  S126     12   X'0144'   X'0A9C'   TCLDELAY
DFHTASK  S127     12   X'0145'   X'0AA8'   MXTDELAY
DFHTASK  S129     12   X'0146'   X'0AB4'   ENQDELAY
DFHTASK  S123     12   X'0147'   X'0AC0'   GNQDELAY
DFHTERM  S133     12   X'0148'   X'0ACC'   LU61WTT
DFHTERM  S134     12   X'0149'   X'0AD8'   LU62WTT
DFHFEPI  S156     12   X'014A'   X'0AE4'   SZWAIT
DFHTASK  S170     12   X'014B'   X'0AF0'   RMITIME
DFHTASK  S171     12   X'014C'   X'0AFC'   RMISUSP
DFHSYNC  S173     12   X'014D'   X'0B08'   SYNCTIME
DFHFILE  S174     12   X'014E'   X'0B14'   RLSWAIT
DFHFILE  S175     12   X'014F'   X'0B20'   RLSCPUT
DFHTASK  S128     12   X'0150'   X'0B2C'   LMDELAY
DFHTASK  S181     12   X'0151'   X'0B38'   WTEXWAIT
DFHTASK  S182     12   X'0152'   X'0B44'   WTCEWAIT
DFHTASK  S183     12   X'0153'   X'0B50'   ICDELAY
DFHTASK  S184     12   X'0154'   X'0B5C'   GVUPWAIT
DFHTEMP  S178     12   X'0155'   X'0B68'   TSSHWAIT
DFHFILE  S176     12   X'0156'   X'0B74'   CFDTWAIT
DFHSYNC  S177     12   X'0157'   X'0B80'   SRVSYWTT
DFHTASK  S191     12   X'0158'   X'0B8C'   RRMSWAIT
DFHTASK  S195     12   X'0159'   X'0B98'   RUNTRWTT
DFHSYNC  S196     12   X'015A'   X'0BA4'   SYNCDLY
DFHSOCK  S241     12   X'015B'   X'0BB0'   SOIOWTT
DFHDATA  S186     12   X'015C'   X'0BBC'   IMSWAIT
DFHDATA  S187     12   X'015D'   X'0BC8'   DB2RDYQW
DFHDATA  S188     12   X'015E'   X'0BD4'   DB2CONWT
DFHDATA  S396     12   X'015F'   X'0BE0'   WMQGETWT
DFHTASK  S253     12   X'0160'   X'0BEC'   JVMTIME
DFHTASK  S254     12   X'0161'   X'0BF8'   JVMSUSP
DFHSOCK  S299     12   X'0162'   X'0C04'   SOOIOWTT
DFHTASK  S192     12   X'0163'   X'0C10'   RQRWAIT
DFHTASK  S193     12   X'0164'   X'0C1C'   RQPWAIT
DFHSYNC  S199     12   X'0165'   X'0C28'   OTSINDWT
DFHTASK  S273     12   X'0166'   X'0C34'   JVMITIME
DFHTASK  S275     12   X'0167'   X'0C40'   JVMRTIME
DFHTASK  S285     12   X'0168'   X'0C4C'   PTPWAIT
DFHTASK  S279     12   X'0169'   X'0C58'   DSMMSCWT
DFHSOCK  S300     12   X'016A'   X'0C64'   ISIOWTT
DFHTASK  S401     12   X'016B'   X'0C70'   JVMTHDWT
DFHDATA  S397     12   X'016C'   X'0C7C'   WMQASRBT
DFHDEST  S403     12   X'016D'   X'0C88'   TDILWTT
DFHDEST  S404     12   X'016E'   X'0C94'   TDELWTT
DFHTASK  S348     12   X'016F'   X'0CA0'   ROMODDLY
DFHTASK  S349     12   X'0170'   X'0CAC'   SOMODDLY
DFHSOCK  S319     12   X'0171'   X'0CB8'   ISALWTT
DFHTERM  S343     12   X'0172'   X'0CC4'   TCALWTT
DFHTASK  S429     12   X'0173'   X'0CD0'   DSAPTHWT
DFHRMI   S001     12   X'0174'   X'0CDC'   RMITOTAL
DFHRMI   S002     12   X'0175'   X'0CE8'   RMIOTHER
DFHRMI   S003     12   X'0176'   X'0CF4'   RMIDB2
DFHRMI   S004     12   X'0177'   X'0D00'   RMIDBCTL
DFHRMI   S005     12   X'0178'   X'0D0C'   RMIEXDLI
DFHRMI   S006     12   X'0179'   X'0D18'   RMIMQM
DFHRMI   S007     12   X'017A'   X'0D24'   RMICPSM
DFHRMI   S008     12   X'017B'   X'0D30'   RMITCPIP
```

Figure 40. Default CICS dictionary entries (part 6)

Performance data sections

Each performance data section is made up of a string of field connectors, followed by one or more performance data records.

All of the performance records produced by a single CICS run have the same format. The default length of the performance records is given in "Performance class data" on page 295. The length of the performance records changes if you add user data at user event monitoring points (EMPs), or if you exclude any system-defined data from the monitoring process.

All of the system-defined data fields in the performance records are described in "Performance class data: listing of data fields" on page 349.

The format of the performance data section is shown in Figure 41.

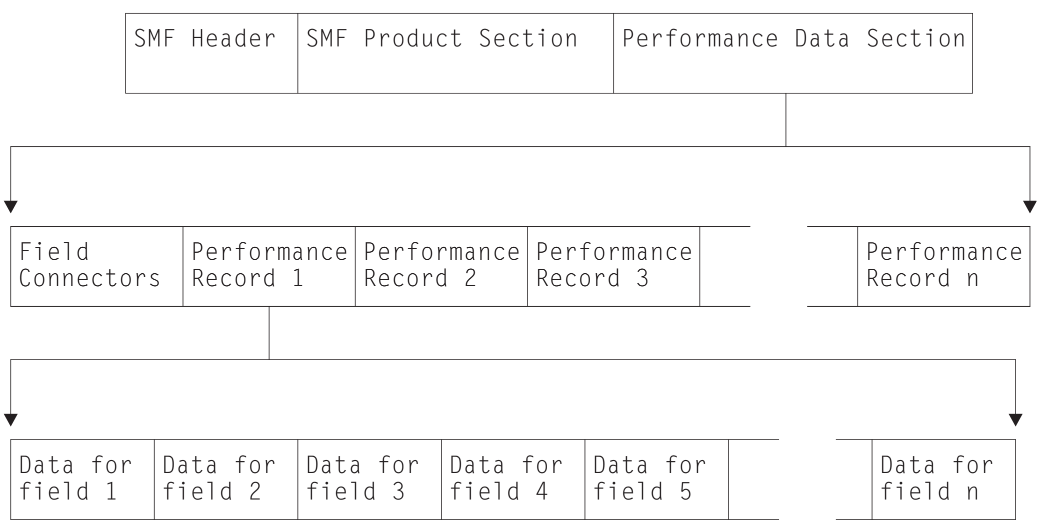


Figure 41. Format of the performance data section

Relationship of the dictionary record to the performance records: field connectors

Following the SMF product section that relates to the performance records, and before the performance records themselves, is a string of **field connectors**. The field connectors connect each performance record field to the dictionary entry that describes it.

The purpose of the field connectors is to tell you which fields are going to occur in the performance records produced by this CICS run. Each field connector corresponds to one field in each of the succeeding performance records. The first field connector corresponds to the first field, the second to the second field, and so on.

Each field connector also corresponds to a single dictionary entry in the associated dictionary record: the connector value is equal to the value of CMODCONN in the corresponding dictionary entry. A useful technique for calculating the offset of a particular dictionary entry is to take the connector, subtract one, and multiply the result by the length of a single dictionary entry.

Thus, the string of field connectors is the key to the dictionary. And without the dictionary, reporting and analysis programs cannot interpret the performance data.

The successive performance records can be regarded as rows in a table, with each column corresponding to one type of field within the records. Each field connector then describes the contents of one column. This view of the data is helpful when designing tabular reports, which are often arranged in this way.

Figure 42 illustrates the relationship between the dictionary record, the field connectors, and the performance records.

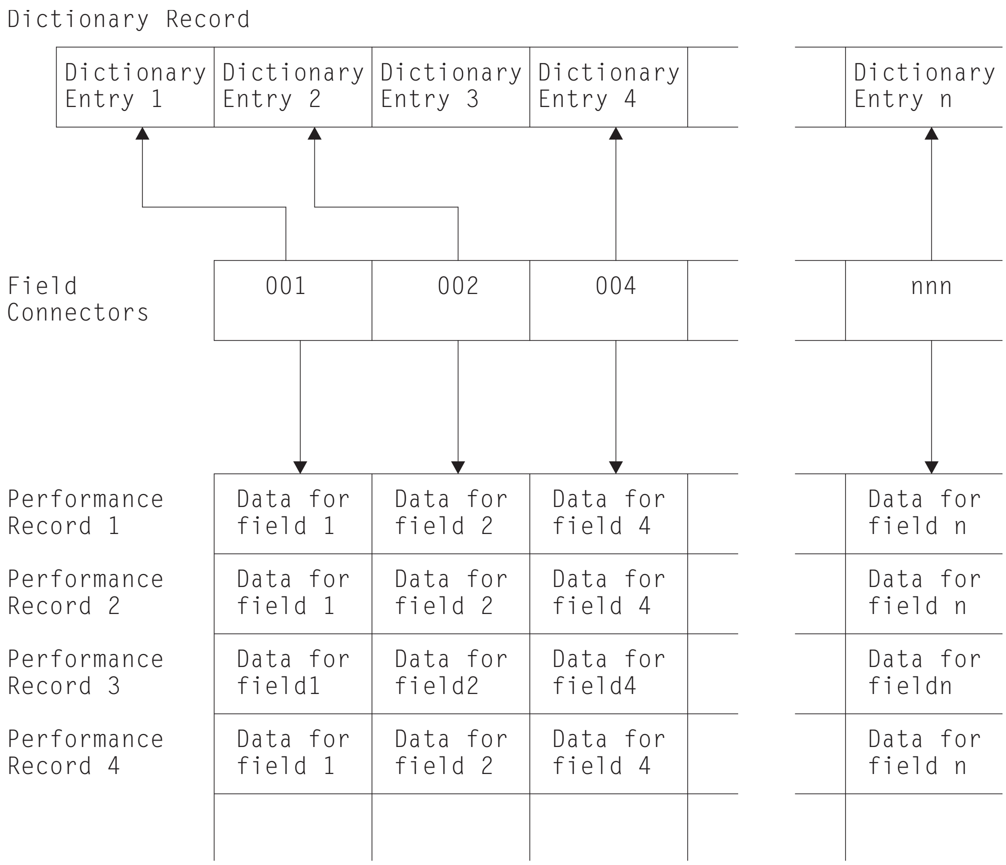


Figure 42. Relationship between the dictionary record and the performance records. In this example, the data that is defined by Dictionary Entry 3 has been excluded, so there is no field connector value for it and it does not appear in the performance records.

How the string of field connectors is constructed

When CICS is initialized, a unique connector value is assigned to every dictionary entry. CICS then examines the MCT entries for this run to see if you have excluded any system-defined performance data. If you have, the offset values for their corresponding dictionary entries are set to X'FFFF'. CICS then constructs a sequence of field connectors that excludes those with offsets of X'FFFF'. In this way, the connectors tell you which system- and user-data fields are going to occur in your performance records for this run. If you have not excluded any system-defined performance data, there is one field connector for every dictionary entry.

Note the difference between field connectors, field identifiers, and field offsets:

Field connectors
link the fields in a performance record with their dictionary entries. They are unique values that are assigned at initialization time. They may, therefore, change from one run of CICS to the next.

Field identifiers
allow you to exclude specific system-defined performance data from being collected during a CICS run. They are unique within a group name and record type, and they do not change between CICS runs. There is more information about field identifiers in Monitoring control table (MCT) in Reference -> System definition.

Field offsets
in the performance record allow you to build a table for fast selection of required fields in your monitoring data processing programs.

Exception data sections

The exception data section contains a single exception record representing one exception condition.

The format of an exception data record (including the SMF header and SMF product section) is shown in Figure 43.

SMF Header	SMF Product Section	Exception Data Section

Figure 43. Format of an SMF exception data record

The format of the exception data section can be mapped by the DSECT MNEXCDS, which you can generate using the DFHMNEXC macro as follows:

```
MNEXCDS DFHMNEXC PREFIX=EXC
```

The label 'MNEXCDS' is the default DSECT name, and EXC is the default PREFIX value, so you could also generate the DSECT by coding

```
DFHMNEXC
```

The MNEXCDS DSECT has the format shown in Figure 44 on page 329.

```
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
EXCMNPOL EQU   X'0004'             POLICY THRESHOLD EXCEEDED
         DS    CL2                 RESERVED
EXCMNTCN DS    CL8                 TRANSACTION CLASS NAME
EXCMNSRV DS    CL8                 SERVICE CLASS NAME
EXCMNRPT DS    CL8                 REPORT CLASS NAME
EXCMNNPX DS    CL20                NETWORK UNIT-OF-WORK PREFIX
EXCMNNSX DS    XL8                 NETWORK UNIT-OF-WORK SUFFIX
EXCMNTRF DS    XL8                 TRANSACTION FLAGS
EXCMNFCN DS    CL4                 TRANSACTION FACILITY NAME
EXCMNCPN DS    CL8                 CURRENT PROGRAM NAME
EXCMNBTR DS    CL4                 BRIDGE TRANSACTION ID
EXCMNURI DS    XL16                RRMS/MVS UNIT OF RECOVERY ID
EXCMNRIL DS    F                   EXCEPTION RESOURCE ID LENGTH
EXCMNRIX DS    XL256               EXCEPTION RESOURCE ID (EXTENDED)
EXCMNNID DS    CL8                 NETWORK ID
EXCMNRLU DS    CL8                 REAL LUNAME
         END OF EXCEPTION RECORD...
```

Figure 44. CICS monitoring exception record DSECT

For further information about exception class data, see “Exception class data” on page 298, which lists all the system-defined data that can be produced by CICS monitoring.

Transaction resource data sections

Each transaction resource data section is made up of one or more transaction resource data records. Transaction resource data records are produced at the end of the transaction for which the data is being collected.

All the transaction resource data records produced by a single CICS run have the same format, with a resource record header followed by a resource data section for each resource being monitored. The records are therefore of variable length, depending on the number of resources for which data is being collected. For example, one transaction might access only one file, but another transaction might access five files and two temporary storage queues.

Each distributed program link adds 32 bytes, each file resource adds 112 bytes, and each temporary storage queue adds a further 120 bytes to a record.

You can collect transaction resource data for up to a maximum of 64 distributed program links, 64 files, and 64 temporary storage queues. The DPL, FILE, and TSQUEUE parameters on the DFHMCT TYPE=INITIAL macro specify the maximum numbers of distributed program links, files, and temporary storage queues for which resource data can be collected for any one transaction. For example, if you specify FILE=10 in the DFHMCT TYPE=INITIAL macro, the file resource data section can have up to 1120 bytes of file resource data.

All the system-defined data fields in the transaction resource monitoring records are described in "Transaction resource class data: Listing of data fields" on page 398.

The format of the transaction resource monitoring records is shown in Figure 45.

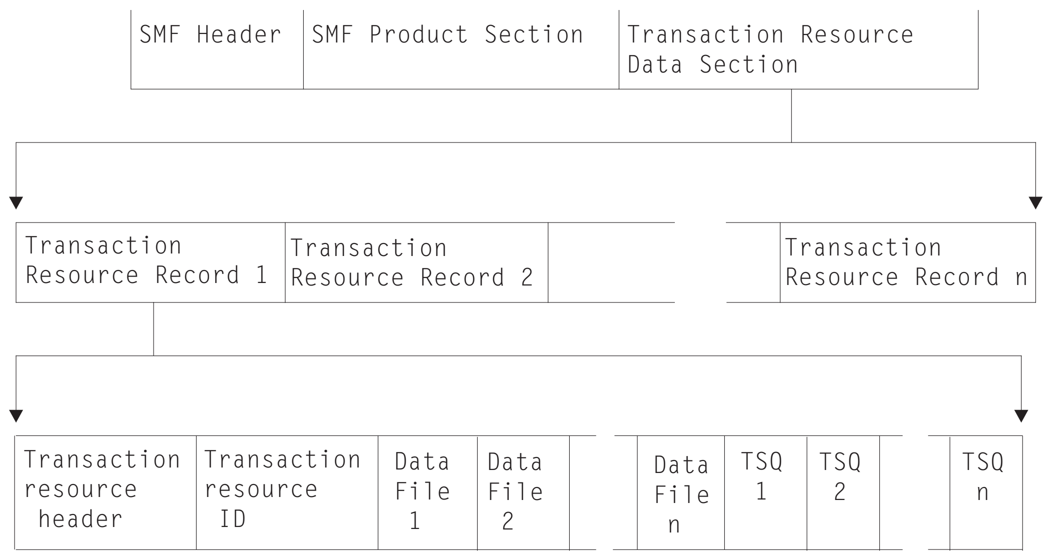


Figure 45. Format of the resource monitoring data section

You can map the transaction resource data section using the DFHMNRDS copybook, details of which are shown in Figure 46 on page 331.

```
DFHMNRDS DSECT  ,                      Monitoring Resource Record
*
         DS     0F                     Fullword allignment
MNR_LENGTH     DS  H                   Length of resource data
MNR_ID_EQUATE  EQU 79                  Monitoring domain id mask
MNR_ID         DC  AL2(MNR_ID_EQUATE)  Monitoring domain id
MNR_VERSION    EQU X'01'               DSECT version mask
MNR_DSECT_VERS DS  CL1                 DSECT version number
               DS  CL3                 Reserved
*
MNR_HEADER     DS  0XL48               Header Data
MNR_HDRLEN     DS  H                   Length of header data
               DS  XL2                 Reserved
               DS  XL8                 Reserved
MNR_TRN        DS  H                   Number of record triplets
               DS  XL2                 Reserved
MNR_ISO        DS  XL4                 Offset to ID data
MNR_ISL        DS  XL2                 Length of ID entry
MNR_ISN        DS  XL2                 Number of ID entries
MNR_FSO        DS  XL4                 Offset to File data
MNR_FSL        DS  XL2                 Length of File entry
MNR_FSN        DS  XL2                 Number of File entries
MNR_TSO        DS  XL4                 Offset to TSQueue data
MNR_TSL        DS  XL2                 Length of TSQueue entry
MNR_TSN        DS  XL2                 Number of TSQueue entries
MNR_DSO        DS  XL4                 Offset to DPL data
MNR_DSL        DS  XL2                 Length of DPL entry
MNR_DSN        DS  XL2                 Number of DPL entries
MNR_HDR_LENGTH EQU *-MNR_HEADER        Header data length
         SPACE ,
MNR_ID_DATA    DSECT                   Identification Data Entry
MNR_ID_TRANID  DS  CL4                 Transaction id
MNR_ID_TERMID  DS  CL4                 Terminal id
MNR_ID_USERID  DS  CL8                 User id
MNR_ID_STYPE   DS  CL4                 Transaction Start type
MNR_ID_START   DS  XL8                 Transaction Start time
MNR_ID_STOP    DS  XL8                 Transaction Stop time
MNR_ID_TASKNO  DS  XL4                 Transaction Sequence Number
MNR_ID_LUNAME  DS  CL8                 VTAM Luname
MNR_ID_PGMNAME DS  CL8                 First program name
MNR_ID_UOW_PX  DS  XL20                Network Unit-of-Work Prefix
MNR_ID_UOW_SX  DS  XL8                 Network Unit-of-Work Suffix
MNR_ID_RSYSID  DS  CL4                 Remote sysid routed to
MNR_ID_TRN_FLAGS DS  XL8               Transaction flags
MNR_ID_FCTYNAME  DS  CL4               Transaction Facility name
MNR_ID_RTYPE     DS  CL4               Resource Record Type
MNR_ID_TERMINFO  DS  0XL4              Terminal Information
MNR_ID_NATURE    DS  XL1               Nature
MNR_ID_NATURE_NOTAPPLIC EQU X'00'          Not applic
MNR_ID_NATURE_TERMINAL  EQU X'01'          Terminal
MNR_ID_NATURE_SESSION   EQU X'02'          Session
MNR_ID_SESSTYPE  DS  XL1               Session Type
MNR_ID_SESSTYPE_NOTAPPLIC EQU X'00'        Not applic
MNR_ID_SESSTYPE_IRC       EQU X'01'        IRC
MNR_ID_SESSTYPE_IRC_XM    EQU X'02'        IRC XM
MNR_ID_SESSTYPE_IRC_XCF   EQU X'03'        IRC XCF
MNR_ID_SESSTYPE_LU61      EQU X'04'        LU61
MNR_ID_SESSTYPE_LU62_SING EQU X'05'        LU62 SINGLE
MNR_ID_SESSTYPE_LU62_PARA EQU X'06'        LU62 PARALLEL
MNR_ID_ACMETH    DS  XL1               Access method
MNR_ID_ACMETH_NOTAPPLIC EQU X'00'          Not applic
MNR_ID_ACMETH_VTAM      EQU X'01'          VTAM
MNR_ID_ACMETH_BSAM      EQU X'03'          BSAM
```

Figure 46. CICS transaction resource monitoring record DSECT (part 1)

```
MNR_ID_ACMETH_TCAM      EQU X'04'          TCAM
MNR_ID_ACMETH_BGAM      EQU X'06'          BGAM
MNR_ID_ACMETH_CONSOLE   EQU X'07'          CONSOLE
MNR_ID_DEVCODE   DS  XL1               Device type code
*                                          See TYPETERM RDO attribute
MNR_ID_TERMCNNM  DS  CL4               Terminal Connection name
MNR_ID_RES_FLAGS DS  0XL4              Resource flags
MNR_ID_RES_FLAG1 DS  XL1               Resource flag 1
MNR_FILE_LIMIT_EXCEEDED    EQU X'80'   Resource File limit exceeded
MNR_TSQUEUE_LIMIT_EXCEEDED EQU X'40'   Resource TSQueue limit exceeded
MNR_DPL_LIMIT_EXCEEDED    EQU X'20'   Resource DPL limit exceeded
               DS  XL3                 Reserved
MNR_ID_ISIPICNM DS XL8                 IPCONN name
               DS  XL8                 Reserved
               DS  XL8                 Reserved
MNR_ID_CLIPADDR         DS CL40        Client IP Address
MNR_ID_ORIGIN_NETWKID   DS CL8         Originating networked
MNR_ID_ORIGIN_APPLID    DS CL8         Originating applid
MNR_ID_ORIGIN_ATT_TIME  DS CL8         Originating task start time
MNR_ID_ORIGIN_TRANNUM   DS CL4         Originating tran seq no
MNR_ID_ORIGIN_TRANID    DS CL4         Originating tran id
MNR_ID_ORIGIN_USERID    DS CL8         Originating userid
MNR_ID_ORIGIN_USER_CORR DS CL64        Originating user data
MNR_ID_ORIGIN_TCPIPSERV DS CL8         Originating TCPIPSERVICE
MNR_ID_ORIGIN_PORTNUM   DS XL4         Originating portnumber
MNR_ID_ORIGIN_CLIPADDR  DS CL40        Originating Client IPaddress
MNR_ID_ORIGIN_CLIPPORT  DS XL4         Originating client portnum
MNR_ID_ORIGIN_TRANFLAG  DS XL8         Originating tran flags
MNR_ID_ORIGIN_FCTYNAME  DS CL8         Originating facility name
MNR_ID_LENGTH  EQU *-MNR_ID_DATA       Identification entry data length
         SPACE ,
MNR_FILE_ENTRY DSECT                   File Entry
MNR_FILE_NAME  DS  CL8                 File name
MNR_FILE_GET   DS  XL8                 File Get time/count
MNR_FILE_PUT   DS  XL8                 File Put time/count
MNR_FILE_BRWSE DS  XL8                 File Browse time/count
MNR_FILE_ADD   DS  XL8                 File Add time/count
MNR_FILE_DEL   DS  XL8                 File Delete time/count
MNR_FILE_TOTAL DS  XL8                 File Total time/count
MNR_FILE_AM_RQ DS  XL4                 File Access Method request count
               DS  XL4                 Reserved
MNR_FILE_IO_WT DS  XL8                 File I/O wait time
MNR_RLS_FILE_IO_WT DS  XL8             RLS File I/O wait time
MNR_CFDT_IO_WT DS  XL8                 CFDT I/O wait time
MNR_FILE_XC_WT DS  XL8                 File exclusive wait
MNR_FILE_VS_WT DS  XL8                 File VSAM string wait
               DS  XL8                 Reserved
MNR_FILE_LEN   EQU *-MNR_FILE_ENTRY    File entry data length
         SPACE ,
MNR_TSQUEUE_ENTRY     DSECT            TSQueue Entry
MNR_TSQUEUE_NAME      DS  CL16         TSQueue Name
MNR_TSQUEUE_GET       DS  XL8          TSQueue Get time/count
MNR_TSQUEUE_PUT_AUX   DS  XL8          TSQueue Put Aux time/count
MNR_TSQUEUE_PUT_MAIN  DS  XL8          TSQueue Put Main time/count
MNR_TSQUEUE_TOTAL     DS  XL8          TSQueue Total time/count
                      DS  XL4          Reserved
MNR_TSQUEUE_GET_ITEML DS  XL4          TSQueue Get Item length
MNR_TSQUEUE_PUT_AUX_ITEML  DS XL4      TSQueue Put Aux Item length
MNR_TSQUEUE_PUT_MAIN_ITEML DS XL4      TSQueue Put Main Item length
                      DS  XL8          Reserved
MNR_TSQUEUE_IO_WT     DS  XL8          TSQueue I/O wait time
MNR_SHR_TSQUEUE_IO_WT DS  XL8          Shared TSQueue I/O wait time
                      DS  XL8          Reserved
MNR_TSQUEUE_GET_SHR   DS  XL8          TSQueue Get Shr time/ct
MNR_TSQUEUE_PUT_SHR   DS  XL8          TSQueue Put Shr time/ct
MNR_TSQUEUE_GET_SHR_ITEML   DS  XL4          TSQueue Get Shr Item len
MNR_TSQUEUE_PUT_SHR_ITEML   DS  XL4          TSQueue Put Shr Item len
MNR_TSQUEUE_LEN EQU *-MNR_TSQUEUE_ENTRY TSQueue entry data length
         SPACE ,
MNR_DPL_ENTRY         DSECT            DPL Entry
MNR_DPL_PROGRAM_NAME  DS  CL8          DPL Program name
MNR_DPL_SYSID         DS  CL4          DPL sysid
                      DS  XL4          Reserved
```

Note: VTAM is now z/OS Communications Server.

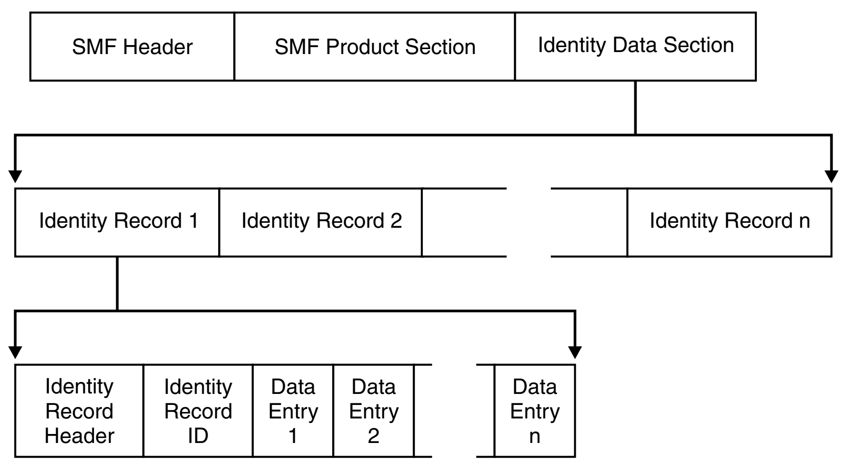
Identity class data sections

Each identity class data section is made up of one or more identity class data records. Identity class data records are produced during transaction detach processing for each transaction that has identity propagation data.

Identity data is constructed using fields that are written only if the data is available, in a similar way to those fields used in the RACF SMF records. Unlike other monitoring SMF 110 records, these records are not compressed. The identity records are buffered (one or more identity records are constructed into a single SMF 110 record) to minimize the number of SMF writes. Any unwritten identity data records remaining in the output buffer are recorded either when the monitoring identity class is set to inactive or when CICS shuts down normally.

The format of the identity class monitoring records is shown in Figure 48.

Figure 48. Format of the identity class data section



The system-defined data fields in the identity class monitoring records are described in "Identity class data: Listing of data fields" on page 405.

You can map the identity class data section using the DFHMNIDS copybook, details of which are shown in Figure 49 on page 335.

```
DFHMNIDS DSECT  ,                      Monitoring Identity Record
*
         DS    0F                      Fullword allignment
MNI_LENGTH     DS  H                   Length of identity data
MNI_ID_EQUATE  EQU 51                  Monitoring domain id mask
MNI_ID         DC  AL2(MNI_ID_EQUATE)  Monitoring domain id
MNI_VERSION    EQU X'01'               DSECT version mask
MNI_DSECT_VERS DS  CL1                 DSECT version number
               DS  CL3                 Reserved
*
MNI_HEADER     DS  0XL32               Header Data
MNI_HDRLEN     DS  H                   Length of header data
               DS  XL2                 Reserved
               DS  XL8                 Reserved
MNI_TRN        DS  H                   Number of record triplets
               DS  XL2                 Reserved
MNI_ISO        DS  XL4                 Offset to ID data
MNI_ISL        DS  XL2                 Length of ID entry
MNI_ISN        DS  XL2                 Number of ID entries
MNI_DSO        DS  XL4                 Offset to Data entry
MNI_DSL        DS  XL2                 Length of Data entry
MNI_DSN        DS  XL2                 Number of Data entries
MNI_HDR_LENGTH EQU *-MNI_HEADER        Header data length
         SPACE ,
MNI_ID_DATA    DSECT                   Identification Data Entry
MNI_ID_TRANID  DS  CL4                 Transaction id
MNI_ID_TERMID  DS  CL4                 Terminal id
MNI_ID_USERID  DS  CL8                 User id
MNI_ID_STYPE   DS  CL4                 Transaction Start type
MNI_ID_START   DS  XL8                 Transaction Start time
MNI_ID_STOP    DS  XL8                 Transaction Stop time
MNI_ID_TASKNO  DS  XL4                 Transaction Sequence Number
MNI_ID_LUNAME  DS  CL8                 VTAM Luname
MNI_ID_PGMNAME DS  CL8                 First program name
MNI_ID_UOW_PX  DS  XL20                Network Unit-of-Work Prefix
MNI_ID_UOW_SX  DS  XL8                 Network Unit-of-Work Suffix
MNI_ID_RSYSID  DS  CL4                 Remote sysid routed to
MNI_ID_TRN_FLAGS DS  XL8               Transaction flags
MNI_ID_FCTYNAME  DS  CL4               Transaction Facility name
MNI_ID_RTYPE     DS  CL4               Resource Record Type
MNI_ID_TERMINFO  DS  0XL4              Terminal Information
MNI_ID_NATURE    DS  XL1               Nature
MNI_ID_NATURE_NOTAPPLIC EQU X'00'          Not applic
MNI_ID_NATURE_TERMINAL  EQU X'01'          Terminal
MNI_ID_NATURE_SESSION   EQU X'02'          Session
MNI_ID_SESSTYPE  DS  XL1               Session Type
MNI_ID_SESSTYPE_NOTAPPLIC EQU X'00'        Not applic
MNI_ID_SESSTYPE_IRC       EQU X'01'        IRC
MNI_ID_SESSTYPE_IRC_XM    EQU X'02'        IRC XM
MNI_ID_SESSTYPE_IRC_XCF   EQU X'03'        IRC XCF
MNI_ID_SESSTYPE_LU61      EQU X'04'        LU61
MNI_ID_SESSTYPE_LU62_SING EQU X'05'        LU62 SINGLE
MNI_ID_SESSTYPE_LU62_PARA EQU X'06'        LU62 PARALLEL
MNI_ID_ACMETH    DS  XL1               Access method
MNI_ID_ACMETH_NOTAPPLIC EQU X'00'          Not applic
MNI_ID_ACMETH_VTAM      EQU X'01'          VTAM
MNI_ID_ACMETH_BSAM      EQU X'03'          BSAM
MNI_ID_ACMETH_TCAM      EQU X'04'          TCAM
MNI_ID_ACMETH_BGAM      EQU X'06'          BGAM
MNI_ID_ACMETH_CONSOLE   EQU X'07'          CONSOLE
MNI_ID_DEVCODE   DS  XL1               Device type code
*                                          See TYPETERM RDO attribute
MNI_ID_TERMCNNM  DS  CL4               Terminal Connection name
               DS  XL4                 Reserved
MNI_ID_ISIPICNM DS XL8                 IPCONN name
               DS  XL8                 Reserved
               DS  XL8                 Reserved
```

```
MNI_ID_CLIPADDR         DS CL40       Client IP Address
MNI_ID_ORIGIN_NETWKID   DS CL8        Originating networkid
MNI_ID_ORIGIN_APPLID    DS CL8        Originating applid
MNI_ID_ORIGIN_ATT_TIME  DS CL8        Originating task start time
MNI_ID_ORIGIN_TRANNUM   DS CL4        Originating tran seq no
MNI_ID_ORIGIN_TRANID    DS CL4        Originating tran id
MNI_ID_ORIGIN_USERID    DS CL8        Originating userid
MNI_ID_ORIGIN_USER_CORR DS CL64       Originating user data
MNI_ID_ORIGIN_TCPIPSERV DS CL8        Originating TCPIPSERVICE
MNI_ID_ORIGIN_PORTNUM   DS XL4        Originating portnumber
MNI_ID_ORIGIN_CLIPADDR  DS CL40       Originating Client IPaddress
MNI_ID_ORIGIN_CLIPPORT  DS XL4        Originating Client portnum
MNI_ID_ORIGIN_TRANFLAG  DS XL8        Originating transaction flags
MNI_ID_ORIGIN_FCTYNAME  DS CL8        Originating facility name
MNI_ID_LENGTH  EQU *-MNI_ID_DATA      Identification entry data length
         SPACE ,
MNI_DATA_ENTRY     DSECT              Data Entry
MNI_ENTRY_IDENT    DS  XL2            Data entry ident
MNI_ENTRY_LENGTH   DS  XL2            Data entry length
MNI_ENTRY_FIELD    DS  0C             Data entry field
```

Figure 49. CICS identity class monitoring record DSECT

Note: VTAM is now z/OS Communications Server.

Clocks and time stamps

In the descriptions of CICS monitoring data, the term **clock** is distinguished from the term **time stamp**.

A **time stamp** is an 8-byte copy of the output of a local store clock (STCK) instruction.

A **clock** consists of three components, arranged in order:

1. **Timer component.** This is a value giving the accumulated time recorded by the clock, expressed in local store clock (STCK) units. For performance class data, the timer component is a 64-bit value. For transaction resource class data, the timer component is a 32-bit value, expressed in units of 16 microseconds. For exception class data, there are no clocks. For more information about timer components, see the TOD clock information in *z/Architecture® Principles of Operation*.
2. **8 reserved bits.**
3. **Period count.** The time recorded by the timer component is accumulated during one or more measurement periods. The period count is a 24-bit value giving the number of measurement periods. The period count runs to 16 777 216.

Neither the timer component of a clock nor its period count are protected against wraparound. The capacity of the clock depends on the class of monitoring data to which the clock applies:

- For performance class data, the clock capacity is only bounded by the capacity of the local store clock, which is several years.
- For transaction resource class data, the clock capacity is about 18 hours.

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).

All times produced in the offline reports are in GMT (Greenwich Mean Time), not local time. Times produced by online reporting can be expressed either in GMT, or in local time, by means of the local date and time offset values from the SMF product section of CICS monitoring SMF type 110 records. The CICS-supplied sample program DFH$MOLS shows an example of this.

Transaction timing fields

The CMF performance class record provides detailed timing information for each transaction as it is processed by CICS. A transaction can be represented by one or more performance class records, depending on the monitoring options that are selected.

The key transaction timing data fields are as follows:

- The Transaction Start time and Stop time represent the start and end of a transaction measurement interval. This is normally the period between transaction attach and detach, but the performance class record might represent a part of a transaction, depending on the monitoring options selected. To calculate the “Transaction Response Time”, subtract the transaction start time from the stop time.
- The Transaction Dispatch time is the time the transaction was dispatched.
- The Transaction Dispatch Wait time is the time the transaction was suspended and waiting for redispatch.
- The Transaction CPU time is the portion of dispatch time when the task is using processor cycles.
- The Transaction Suspend time is the total time the task was suspended. This includes all task suspend (wait) time and includes the following fields:
  - The wait time for redispatch (dispatch wait).
  - The wait time for first dispatch (first dispatch delay). This delay is broken down further into the following fields:
    - First dispatch delay caused by TRANCLASS limits.
    - First dispatch delay caused by MXT limits.
  - The total I/O wait and other wait times.

The CMF performance class record also provides a more detailed breakdown of the transaction suspend (wait) time into separate data fields. These comprise the following fields:

Table 28. Performance class wait (suspend) fields

Group Name	Field ID	Field Name	Description
DFHTERM	009	TCIOWTT	Terminal I/O wait time

Table 28. Performance class wait (suspend) fields (continued)

Group Name	Field ID	Field Name	Description
DFHJOUR	010	JCIOWTT	Journal I/O wait time
DFHTEMP	011	TSIOWTT	Temporary storage I/O wait time
DFHFILE	063	FCIOWTT	File I/O wait time
DFHTERM	100	IRIOWTT	Interregion I/O wait time
DFHDEST	101	TDIOWTT	Transient data I/O wait time
DFHTASK	123	GNQDELAY	Global ENQ delay time
DFHTASK	128	LMDELAY	Lock Manager delay time
DFHTASK	129	ENQDELAY	Local ENQ delay time
DFHTERM	133	LU61WTT	LU 6.1 I/O wait time
DFHTERM	134	LU62WTT	LU 6.2 I/O wait time
DFHFEPI	156	SZWAIT	FEPI suspend time
DFHTASK	171	RMISUSP	Resource manager interface (RMI) suspend time
DFHFILE	174	RLSWAIT	RLS File I/O wait time
DFHFILE	176	CFDTWAIT	Coupling Facility data tables server I/O wait time
DFHSYNC	177	SRVSYWTT	Coupling Facility data tables server sync point and resynchronization wait time
DFHTEMP	178	TSSHWAIT	Shared temporary storage I/O wait time
DFHTASK	181	WTEXWAIT	EXEC CICS WAIT EXTERNAL wait time
DFHTASK	182	WTCEWAIT	EXEC CICS WAITCICS and WAIT EVENT wait time
DFHTASK	183	ICDELAY	Interval Control delay time
DFHTASK	184	GVUPWAIT	Dispatchable Wait wait time
DFHDATA	186	IMSWAIT	IMS DBCTL wait time
DFHDATA	187	DB2RDYQW	DB2 ready queue wait time
DFHDATA	188	DB2CONWT	DB2 connection time
DFHTASK	191	RRMSWAIT	RRMS/MVS indoubt wait time
DFHTASK	192	RQRWAIT	Request Receiver wait time
DFHTASK	193	RQPWAIT	Request Processor wait time
DFHTASK	195	RUNTRWTT	CICS BTS run process/activity synchronous wait time
DFHSYNC	196	SYNCDLY	Sync point delay time
DFHSOCK	241	SOIOWTT	Inbound Socket I/O wait time
DFHTASK	247	DSCHMDLY	CICS change TCB mode delay time
DFHTASK	250	MXTOTDLY	CICS L8 and L9 mode open TCB delay time
DFHTASK	254	JVMSUSP	JVM suspend time
DFHTASK	268	DSTCBMWT	TCB mismatch wait time
DFHTASK	279	DSMMSCWT	MVS storage constraint wait time
DFHTASK	281	MAXSTDLY	CICS SSL TCB delay time
DFHTASK	282	MAXXTDLY	CICS XP TCB delay time

Table 28. Performance class wait (suspend) fields (continued)

Group Name	Field ID	Field Name	Description
DFHTASK	283	MAXTTDLY	CICS JVM server thread TCB delay time
DFHTASK	285	PTPWAIT	3270 bridge partner wait time
DFHSOCK	299	SOOIOWT	Outbound Socket I/O wait time
DFHSOCK	300	ISIOWTT	IS I/O wait time
DFHSOCK	319	ISALWTT	IPIC session allocation wait time
DFHTERM	343	TCALWTT	MRO, LU6.1, and LU6.2 session allocation wait time
DFHDATA	396	WMQGETWT	MQ GETWAIT wait time
DFHTASK	401	JVMTHDWT	JVM server thread wait time
DFHDEST	403	TDILWTT	Transient Data intrapartition lock wait time
DFHDEST	404	TDELWTT	Transient Data extrapartition lock wait time
DFHFILE	426	FCXCWTT	File control wait time for exclusive control of a VSAM control interval
DFHFILE	427	FCVSWTT	File control wait time for a VSAM string
DFHTASK	429	DSAPTHWT	Dispatcher allocate pthread wait time

Transaction response time

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

Figure 50 shows the relationship of dispatch time, suspend time, and CPU time with the response time.

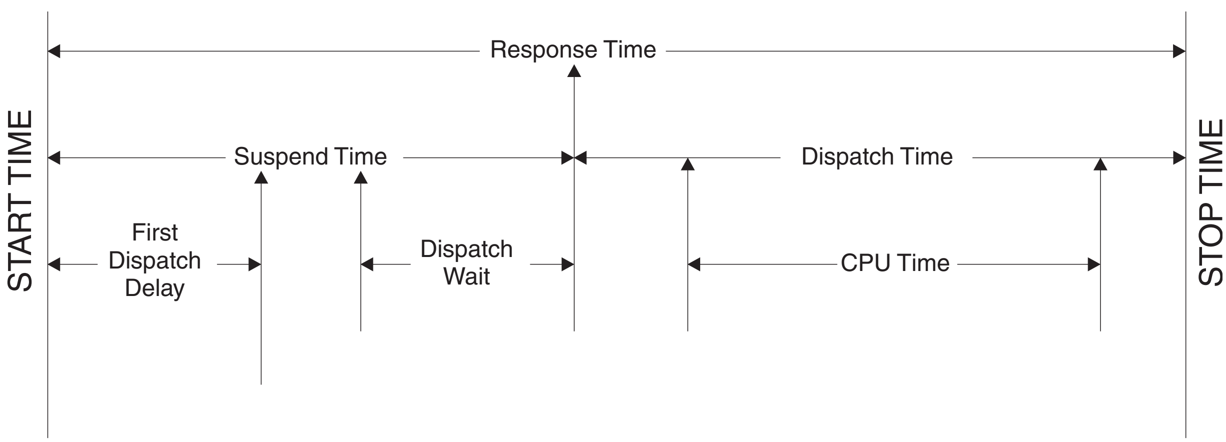


Figure 50. Response time relationships

Transaction dispatch time and CPU time

The transaction total dispatch time field USRDISPT, field 007 in group DFHTASK, is the total **elapsed** time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.

The transaction total CPU time field USRCPUT, field 008 in group DFHTASK, is the total **processor** time during which the user task was dispatched by the CICS dispatcher domain on each CICS TCB under which the task executed.

For both these fields, the time recorded in the field can be associated with any of the TCB modes which are managed by the CICS dispatcher in the current CICS release. These include open TCBs, such as L8 mode TCBs, as well as non-open TCBs, such as the QR TCB. Be aware that for each CICS release, new TCB modes might be added or obsolete TCB modes might be removed, particularly in the case of the open TCB modes. You should always check the performance data field descriptions in the current release documentation to see which TCB modes are applicable. The field descriptions are listed in "Performance data in group DFHTASK" on page 371.

If you want to calculate a transaction's ratio of accumulated CPU time to accumulated dispatch time (CPU/DISP ratio) for the QR TCB, use fields 255 (QRDISPT) and 256 (QRCPUT) in group DFHTASK. These fields show the elapsed time and processor time during which the user task was dispatched on the QR TCB only.

The CPU/DISP ratio for an individual task should always be considered in the context of other activity in the CICS region. The Dispatcher TCB Modes report (see "Dispatcher TCB Modes report" on page 818) which is provided by the sample statistics program DFH0STAT includes a calculation of the CPU/DISP ratio for the QR TCB for the whole CICS region.

Transaction wait (suspend) times

The CMF performance class record provides a breakdown of the transaction suspend (wait) time into separate data fields. You can use these to calculate various wait times.

The performance data fields listed in Table 29 record the elapsed time spent waiting for specific types of I/O operation. For example, field 009 records the elapsed time waiting for terminal I/O.

The elapsed time includes the time for the I/O operation, the time for the access method to complete the outstanding event control block, and the subsequent time until the waiting CICS transaction is redispatched.

Table 29. Performance class wait (suspend) fields

Group Name	Field ID	Field Name	Description
DFHTERM	009	TCIOWTT	Terminal I/O wait time
DFHJOUR	010	JCIOWTT	Journal I/O wait time
DFHTEMP	011	TSIOWTT	Temporary storage I/O wait time
DFHFILE	063	FCIOWTT	File I/O wait time
DFHTERM	100	IRIOWTT	Interregion I/O wait time
DFHDEST	101	TDIOWTT	Transient data I/O wait time
DFHTASK	123	GNQDELAY	Global ENQ delay time
DFHTASK	128	LMDELAY	Lock Manager delay time
DFHTASK	129	ENQDELAY	Local ENQ delay time

Table 29. Performance class wait (suspend) fields (continued)

Group Name	Field ID	Field Name	Description
DFHTERM	133	LU61WTT	LU 6.1 I/O wait time
DFHTERM	134	LU62WTT	LU 6.2 I/O wait time
DFHFEPI	156	SZWAIT	FEPI suspend time
DFHTASK	171	RMISUSP	Resource manager interface (RMI) suspend time
DFHFILE	174	RLSWAIT	RLS File I/O wait time
DFHFILE	176	CFDTWAIT	Coupling Facility data tables server I/O wait time
DFHSYNC	177	SRVSYWTT	Coupling Facility data tables server sync point and resynchronization wait time
DFHTEMP	178	TSSHWAIT	Shared temporary storage I/O wait time
DFHTASK	181	WTEXWAIT	EXEC CICS WAIT EXTERNAL wait time
DFHTASK	182	WTCEWAIT	EXEC CICS WAITCICS and WAIT EVENT wait time
DFHTASK	183	ICDELAY	Interval Control delay time
DFHTASK	184	GVUPWAIT	Dispatchable Wait wait time
DFHDATA	186	IMSWAIT	IMS DBCTL wait time
DFHDATA	187	DB2RDYQW	DB2 ready queue wait time
DFHDATA	188	DB2CONWT	DB2 connection time
DFHTASK	191	RRMSWAIT	RRMS/MVS indoubt wait time
DFHTASK	192	RQRWAIT	Request Receiver wait time
DFHTASK	193	RQPWAIT	Request Processor wait time
DFHTASK	195	RUNTRWTT	CICS BTS run process/activity synchronous wait time
DFHSYNC	196	SYNCDLY	Sync point delay time
DFHSOCK	241	SOIOWTT	Inbound Socket I/O wait time
DFHTASK	247	DSCHMDLY	CICS change TCB mode delay time
DFHTASK	250	MXTOTDLY	CICS L8 and L9 mode open TCB delay time
DFHTASK	254	JVMSUSP	JVM suspend time
DFHTASK	268	DSTCBMWT	TCB mismatch wait time
DFHTASK	279	DSMMSCWT	MVS storage constraint wait time
DFHTASK	281	MAXSTDLY	CICS SSL TCB delay time
DFHTASK	282	MAXXTDLY	CICS XP TCB delay time
DFHTASK	283	MAXTTDLY	CICS JVM server thread TCB delay time
DFHTASK	285	PTPWAIT	3270 bridge partner wait time
DFHSOCK	299	SOOIOWT	Outbound Socket I/O wait time
DFHSOCK	300	ISIOWTT	IS I/O wait time
DFHSOCK	319	ISALWTT	IPIC session allocation wait time
DFHTERM	343	TCALWTT	MRO, LU6.1, and LU6.2 session allocation wait time
DFHDATA	396	WMQGETWT	MQ GETWAIT wait time
DFHTASK	401	JVMTHDWT	JVM server thread wait time

Table 29. Performance class wait (suspend) fields (continued)

Group Name	Field ID	Field Name	Description
DFHDEST	403	TDILWTT	Transient Data intrapartition lock wait time
DFHDEST	404	TDELWTT	Transient Data extrapartition lock wait time
DFHFILE	426	FCXCWTT	File control wait time for exclusive control of a VSAM control interval
DFHFILE	427	FCVSWTT	File control wait time for a VSAM string
DFHTASK	429	DSAPTHWT	Dispatcher allocate pthread wait time

Figure 51 shows an example of the relationship between a typical transaction wait time field, and the transaction's suspend time, dispatch time, processor, and dispatch wait time fields. The transaction has a period of suspend time, which is the time between two periods of dispatch and processor time. The period of suspend time is equal to the total of all the relevant wait times. The period of suspend time includes the dispatch wait, which ends when the suspend time ends and the dispatch and processor time starts.

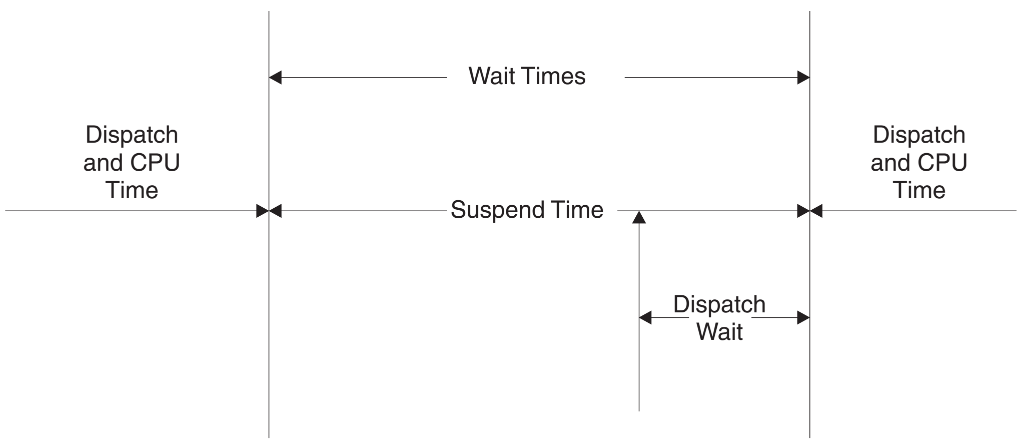


Figure 51. Wait (suspend) time relationships

You can use the CMF suspend time and wait time measurements to perform accurate calculations on the suspend time. For example, to calculate the total I/O wait time, add the values of the fields in the following list:

- Terminal control I/O wait
- Temporary storage I/O wait
- Shared temporary storage I/O wait
- Transient data I/O wait
- Journal (MVS logger) I/O wait
- File control I/O wait
- RLS file I/O wait
- Coupling Facility data table I/O wait
- Inbound Socket I/O wait
- IS I/O wait time
- Outbound Socket I/O wait
- Interregion (MRO) I/O wait

- LU 6.1 TC I/O wait
- LU 6.2 TC I/O wait
- FEPI I/O wait

To calculate the total other wait time, add the values of the fields in the following list:

- First dispatch delay. This field includes the MXT and TRANCLASS first dispatch delay fields.
- Local ENQ delay
- Global ENQ delay
- Interval control delay
- Lock manager delay
- Wait external wait
- EXEC CICS WAITCICS and EXEC CICS WAIT EVENT wait
- CICS BTS run synchronous wait
- CFDT server synchronous wait
- Request Receiver wait time
- Request Processor wait time
- Syncpoint delay time
- CICS L8 and L9 mode open TCB delay time
- CICS SSL TCB delay time
- CICS JVM server thread TCB delay time
- CICS XP TCB delay time
- CICS change-TCB mode delay time
- RRMS/MVS wait
- 3270 bridge partner wait
- TCB mismatch wait time
- JVM server thread wait time
- MVS storage constraint wait time
- Intrapartition transient data lock wait time
- Extrapartition transient data lock wait time
- File control wait time for exclusive control of a VSAM control interval
- File control wait time for a VSAM string
- IPIC session allocation wait time
- MRO, LU6.1, and LU6.2 session allocation wait time
- Dispatchable waits wait
- Dispatcher allocate pthread wait time

To determine the uncaptured wait time, use the following calculation:

```
Uncaptured wait time =  (Suspend - (total I/O wait time + total other wait time))
```

The CMF performance class data also provides the following important transaction timing measurements:

- The Program load time is the program fetch time (dispatch time) for programs invoked by the transaction. See “Program load time” on page 343.

- The Exception wait time is the accumulated time from the exception conditions as measured by the CMF exception class records. For more information, see “Exception class data: listing of data fields” on page 393.
- The RMI elapsed time is the elapsed time the transaction spent in all Resource Managers invoked by the transaction using the Resource Manager Interface (RMI). See “RMI elapsed and suspend time” on page 344.
- The JVM elapsed time is the elapsed time the transaction spent in the Java Virtual Machine (JVM) for the Java programs invoked by the transaction. See “JVM elapsed time and suspend time” on page 344.
- The JVM initialization elapsed time is the elapsed time the transaction spent initializing the Java Virtual Machine (JVM) environment for all the Java programs invoked by the transaction.
- The Syncpoint elapsed time is the elapsed time the transaction spent processing a sync point. See “Syncpoint elapsed time” on page 345.

Program load time

The program load time is the program fetch time (dispatch time) for programs invoked by a transaction.

Figure 52 shows the relationship between the program load time (field id 115 in group DFHPROG) and the dispatch time and the suspend time (field ids 7 and 14 in group DFHTASK).

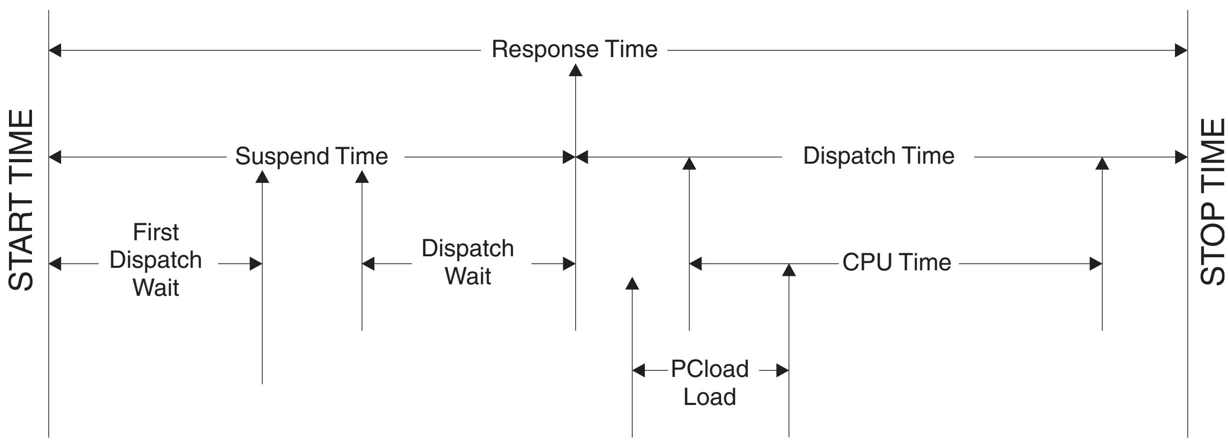


Figure 52. Program load time

The response time of the transaction is the total time from the transaction start time to the transaction stop time. The response time can be subdivided into the following two periods:

- Suspend time.

  The suspend time includes the first dispatch delay, which begins at the transaction start time and ends partway into the suspend time. The suspend time also includes the dispatch wait, which begins further on into the suspend time, and ends when the suspend time ends and the dispatch time begins.
- Dispatch time.

  The dispatch time includes the CPU time, which begins some time after the start of the dispatch time, and ends some time before the dispatch time ends. In this figure, the dispatch time also includes the program load time. The program load time begins after the start of the dispatch time, and overlaps with the first part of the CPU time.

RMI elapsed and suspend time

The RMI elapsed time and suspend time fields provide an insight into the amount of time that a transaction spends in the CICS resource manager interface (RMI).

Figure 53 shows the relationship between the RMI elapsed time (field id 170 in group DFHTASK) and the suspend time (field id 171 in group DFHTASK).

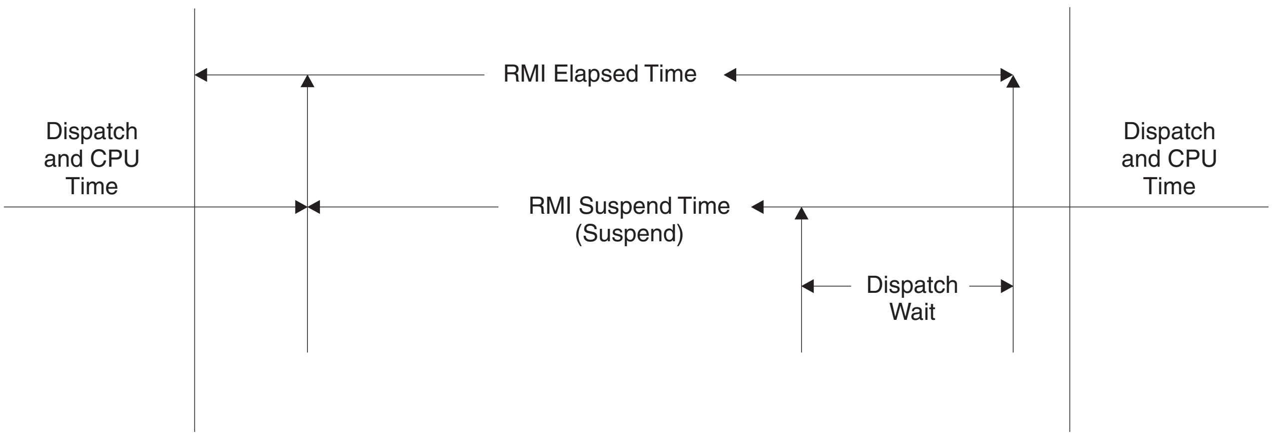


Figure 53. RMI elapsed and suspend time

The RMI elapsed time includes part of a period of dispatch and CPU time at the start. This period of time began before the RMI elapsed time. When the dispatch and CPU time ends, the RMI suspend time begins. The RMI suspend time includes a dispatch wait at the end. When the dispatch wait ends, another period of dispatch and CPU time begins. Shortly afterwards, the RMI elapsed time ends, while the dispatch and CPU time continues. The RMI elapsed time therefore includes parts of two periods of dispatch and CPU time, with an intervening period of RMI suspend time.

The DB2 wait, the DB2 connection wait, and the DB2 readyq wait time fields, as well as the IMS wait and MQ GETWAIT wait time fields are included in the RMI suspend time.

JVM elapsed time and suspend time

The JVM elapsed and suspend time fields provide an insight into the amount of time that a transaction spends in a Java Virtual Machine (JVM).

JVMTIME and JVMSUSP fields

Care must be taken when using the JVM elapsed time field JVMTIME (group name DFHTASK, field id: 253) and JVM suspend time field JVMSUSP (group name DFHTASK, field id: 254) in any calculation with other CMF timing fields. This is because of the likelihood of double accounting other CMF timing fields in the performance class record within the JVM time fields. For example, if a Java application program invoked by a transaction issues a read file (non-RLS) request using the Java API for CICS (JCICS) classes, the file I/O wait time will be included in both the file I/O wait time field (group name DFHFILE, field id: 063), and the transaction suspend time field (group name DFHTASK, field id: 014), as well as the JVM suspend time field.

JCICS requests

The number of Java API for CICS (JCICS) requests issued by the user task is included in the CICS OO foundation class request count field (group name: DFHCICS, field id: 025).

Syncpoint elapsed time

The syncpoint elapsed time is the elapsed time the transaction spent processing a syncpoint.

Figure 54 shows the relationship between the syncpoint elapsed time (field 173 in group DFHSYNC) and the suspend time (field 14 in group DFHTASK).

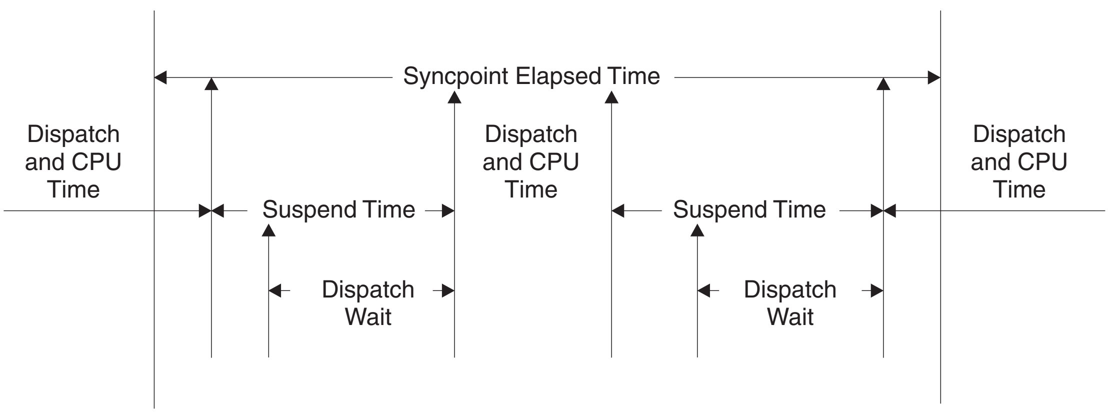


Figure 54. Syncpoint elapsed time

The syncpoint elapsed time begins during a period of dispatch and CPU time. A period of suspend time follows, which includes a dispatch wait at the end. When the dispatch wait and the suspend time end, there is another period of dispatch and CPU time. When this period ends, another period of suspend time begins, which includes another dispatch wait. When the dispatch wait and the suspend time end, another period of dispatch and CPU time begins. Shortly afterward, the syncpoint elapsed time ends, while the period of dispatch and CPU time continues. Therefore, the syncpoint elapsed time in this example includes two complete periods of suspend time.

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 following calculations refer to the middle 4 bytes of each 8 byte start/stop time field. Bit 47 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.

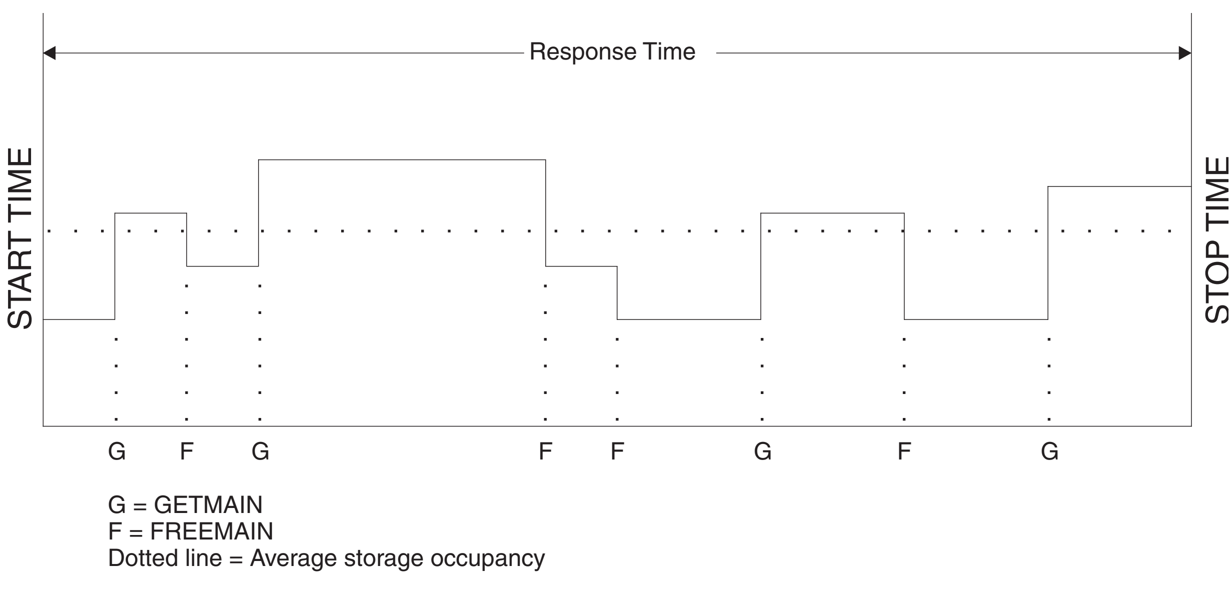


Figure 55. Storage occupancy

Program storage

The level of program storage that is 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. 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 56 shows the relationships between the high-water mark data fields that show the maximum amounts of program storage in use by the user task. The PCSTGHWM field (id 087) shows the maximum amount of program storage in use by the task both above and below 16 MB. The PC31AHWM (139) and PC24BHWM (108) fields are subsets of PCSTGHWM, and show the maximum amounts in use above and below 16M, respectively. Further subset fields show the maximum amounts of storage in use by the task in each of the CICS dynamic storage areas (DSAs).

Note:

1. The total of the values for all the subsets in a superset might not be equal to the value for the superset. For example, the value of PC31AHWM plus the value of PC24BHWM might not be 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.
2. If a task loads the same program several times, the program storage data fields might not reflect the true high-water mark of program storage used by the task. The fields are incremented each time the LOAD command is issued, but if the program has already been loaded by the task, the existing copy of the program is used, meaning that only one copy of the program exists in storage. Because of this, for tasks that repeatedly load the same program, the data in the fields PCSTGHWM, PC24BHWM, PC31RHWM, PC31AHWM, PC31CHWM, PC24CHWM, PC24SHWM, PC31SHWM and PC24RHWM should be used with caution.
3. When you disable an exit program, the program can be released, and the program storage fields might be decremented. If the program was not linked to within the lifetime of the current task, then the program storage fields are decremented unnecessarily, which means you might not get a true value and should proceed with caution.

The high-water mark fields and program storage fields are described in detail in "Performance data in group DFHSTOR" on page 367.

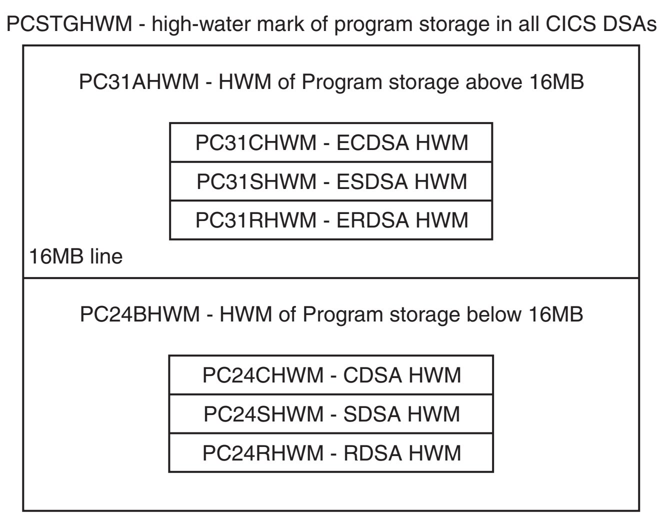


Figure 56. Relationships between the high-water mark program storage data fields

Chapter 30. Monitoring class data: listing of data fields

A list of data fields for exception class data, identity class data, transaction resource class data, and system-defined performance class data.

Performance class data: listing of data fields

The performance class data is listed in this section 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.

Related concepts:

"Performance class data" on page 295
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. CICS writes at least one performance monitoring record for each transaction.

Performance data in group DFHCBTS

Descriptions of the performance data fields in the DFHCBTS group, including the numeric identifier, type, and size of each field.

200 (TYPE-C, 'PRCSNAME', 36 BYTES)
The name of the CICS business transaction service (BTS) process of which the user task formed part.

201 (TYPE-C, 'PRCSTYPE', 8 BYTES)
The process-type of the CICS BTS process of which the user task formed part.

202 (TYPE-C, 'PRCSID', 52 BYTES)
The CICS-assigned identifier of the CICS BTS root activity that the user task implemented.

203 (TYPE-C, 'ACTVTYID', 52 BYTES)
The CICS-assigned identifier of the CICS BTS activity that the user task implemented.

204 (TYPE-C, 'ACTVTYNM', 16 BYTES)
The name of the CICS BTS activity that the user task implemented.

205 (TYPE-A, 'BARSYNCT', 4 BYTES)
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity synchronously.

206 (TYPE-A, 'BARASYCT', 4 BYTES)
The number of CICS BTS run process, or run activity, requests that the user task made in order to execute a process or activity asynchronously.

207 (Type-A, 'BALKPACT', 4 BYTES)
The number of CICS BTS link process, or link activity, requests that the user task issued.

208 (TYPE-A, 'BADPROCT', 4 BYTES)
The number of CICS BTS define process requests issued by the user task.

209 (TYPE-A, 'BADACTCT', 4 BYTES)
The number of CICS BTS define activity requests issued by the user task.

210 (TYPE-A, 'BARSPACT', 4 BYTES)
The number of CICS BTS reset process and reset activity requests issued by the user task.

211 (TYPE-A, 'BASUPACT', 4 BYTES)
The number of CICS BTS suspend process, or suspend activity, requests issued by the user task.

212 (TYPE-A, 'BARMPACT', 4 BYTES)
The number of CICS BTS resume process, or resume activity, requests issued by the user task.

213 (TYPE-A, 'BADCPACT', 4 BYTES)
The number of CICS BTS delete activity, cancel process, or cancel activity, requests issued by the user task.

214 (TYPE-A, 'BAACQPCT', 4 BYTES)
The number of CICS BTS acquire process, or acquire activity, requests issued by the user task.

215 (Type-A, 'BATOTPCT', 4 BYTES)
Total number of CICS BTS process and activity requests issued by the user task.

216 (TYPE-A, 'BAPRDCCT', 4 BYTES)
The number of CICS BTS delete, get, move, or put, container requests for process data containers issued by the user task.

217 (TYPE-A, 'BAACDCCT', 4 BYTES)
The number of CICS BTS delete, get, move, or put, container requests for current activity data containers issued by the user task.

218 (Type-A, 'BATOTCCT', 4 BYTES)
Total number of CICS BTS delete, get, move, or put, process container and activity container requests issued by the user task.

219 (TYPE-A, 'BARATECT', 4 BYTES)
The number of CICS BTS retrieve-reattach event requests issued by the user task.

220 (TYPE-A, 'BADFIECT', 4 BYTES)
The number of CICS BTS define-input event requests issued by the user task.

221 (TYPE-A, 'BATIAECT', 4 BYTES)
The number of CICS BTS DEFINE TIMER EVENT, CHECK TIMER EVENT, DELETE TIMER EVENT, and FORCE TIMER EVENT requests issued by the user task.

222 (TYPE-A, 'BATOTECT', 4 BYTES)
Total number of CICS BTS event-related requests issued by the user task.

Performance data in group DFHCHNL

Descriptions of the performance data fields in the DFHCHNL group, including the numeric identifier, type, and size of each field.

321 (TYPE-A, 'PGTOTCCT', 4 BYTES)
The number of CICS requests for channel containers issued by the user task.

322 (TYPE-A, 'PGBRWCCT', 4 BYTES)
The number of CICS browse requests for channel containers issued by the user task.

323 (TYPE-A, 'PGGETCCT', 4 BYTES)
The number of GET CONTAINER and GET64 CONTAINER requests for channel containers issued by the user task.

324 (TYPE-A, 'PGPUTCCT', 4 BYTES)
The number of PUT CONTAINER and PUT64 CONTAINER requests for channel containers issued by the user task.

325 (TYPE-A, 'PGMOVCCT', 4 BYTES)
The number of MOVE CONTAINER requests for channel containers issued by the user task.

326 (TYPE-A, 'PGGETCDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the GET CONTAINER CHANNEL and GET64 CONTAINER CHANNEL commands issued by the user task.

327 (TYPE-A, 'PGPUTCDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the PUT CONTAINER CHANNEL and PUT64 CONTAINER CHANNEL commands issued by the user task.

328 (TYPE-A, 'PGCRECCT', 4 BYTES)
The number of containers created by MOVE, PUT CONTAINER, and PUT64 CONTAINER requests for channel containers issued by the user task.

329 (TYPE-A, 'PGCSTHWM', 4 BYTES)
Maximum amount (high-water mark), in bytes, of container storage allocated to the user task.

Performance data in group DFHCICS

Descriptions of the performance data fields in the DFHCICS group, including the numeric identifer, type, and size of each field.

005 (TYPE-T, 'START', 8 BYTES)
Start time of measurement interval, which is one of the following times:
- 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 or the monitoring options MNCONV, MNSYNC, or FREQUENCY.

For more information, see Clocks and time stamps in Reference -> Monitoring.

Note: Response time = STOP - START. For more information, see Transaction response time in Reference -> Monitoring.

006 (TYPE-T, 'STOP', 8 BYTES)
Finish time of measurement interval, which is one of the following times:
- The time at which the user task was detached.
- 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 in Reference -> Monitoring.

Note: Response time = STOP - START. For more information, see Transaction response time in Reference -> Monitoring.

025 (TYPE-A, 'CFCAPICT', 4 BYTES)
Number of CICS OO foundation class requests, including the Java API for CICS (JCICS) classes, issued by the user task.

089 (TYPE-C, 'USERID', 8 BYTES)
User identification at task creation. This identification 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', 12 BYTES)
Accumulated data for exception conditions. The timer component of the clock contains the total elapsed time for which the user waited on exception conditions. The period count equals the number of exception conditions that have occurred for this task. For more information on exception conditions, see Exception class data: Listing of data fields in Reference -> Monitoring. For more information on clocks, see Clocks and time stamps in Reference -> Monitoring.

Note: The performance class data field 'EXWTTIME' is updated when exception conditions are encountered even when the exception class is inactive.

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 sync point

T Record output for the end of a task.

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

This field also includes the connection name (SYSID) of the remote system to which this transaction was routed when using the CRTE routing transaction. The field is null for those CRTE transactions that establish or cancel the transaction routing session.

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

131 (TYPE-A, 'PERRECNT', 4 BYTES)
The number of performance class records written by the CICS Monitoring Facility (CMF) for the user task.

167 (TYPE-C, 'SRVCLASS', 8 BYTES)
The z/OS Workload Manager (WLM) service class for this transaction. This field is null if no transaction classification rules are defined for CICS subsystems in the active z/OS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

168 (TYPE-C, 'RPTCLASS', 8 BYTES)
The z/OS Workload Manager (WLM) report class for this transaction. This field is null if no transaction classification rules are defined for CICS subsystems in the active z/OS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

351 (TYPE-C, 'OADID', 64 BYTES)
The adapter identifier added to the origin data by the adapter. This field is blank if the task was not started by using an adapter, or if it was and the adapter did not set this value.

352 (TYPE-C, 'OADATA1', 64 BYTES)
The data added to the origin data by the adapter. This field is blank if the task was not started by using an adapter, or if it was and the adapter did not set this value.

353 (TYPE-C, 'OADATA2', 64 BYTES)
The data added to the origin data by using the adapter. This field is blank if the task was not started by using an adapter, or if it was and the adapter did not set this value.

354 (TYPE-C, 'OADATA3', 64 BYTES)
The data added to the origin data by the adapter. This field is blank if the task was not started by using an adapter, or if it was and the adapter did not set this value.

359 (TYPE-C 'ONETWKID', 8 BYTES
The network identifier from which this work request (transaction) originated.

360 (TYPE-C, 'OAPPLID', 8 BYTES)
The APPLID of the CICS region in which this work request (transaction) originated; for example, the region in which the CWXN task ran.

361 (TYPE-T, 'OSTART', 8 BYTES)
The time at which the originating task, for example the CWXN task, was started.

362 (TYPE-P, 'OTRANNUM', 4 BYTES)
The number of the originating task; for example, the CWXN task.

363 (TYPE-C, 'OTRAN', 4 BYTES)
The transaction ID (TRANSID) of the originating task; for example, the CWXN task.

364 (TYPE-C, 'OUSERID', 8 BYTES)
The originating Userid-2 or Userid-1, for example from CWBA, depending on the originating task.

365 (TYPE-C, 'OUSERCOR', 64 BYTES)
The originating user correlator.

366 (TYPE-C, 'OTCPSVCE', 8 BYTES)
The name of the originating TCPIPSERVICE.

367 (TYPE-A, 'OPORTNUM', 4 BYTES)
The port number used by the originating TCPIPSERVICE.

369 (TYPE-A, 'OCLIPORT', 4 BYTES)
The TCP/IP port number of the originating client or Telnet client.

370 (TYPE-A, 'OTRANFLG', 8 BYTES)
Originating transaction flags, a string of 64 bits used for signaling transaction definition and status information:

Byte 0
The facility-type of the originating transaction:

Bit 0 None (X'80')

Bit 1 Terminal (X'40')

Bit 2 Surrogate (X'20')

Bit 3 Destination (X'10')

Bit 4 3270 bridge (X'08')

Bit 5 Reserved

Bit 6 Reserved

Bit 7 Reserved

Byte 1
Transaction identification information:

Bit 0 System transaction (x'80')

Bit 1 Mirror transaction (x'40')

Bit 2 DPL mirror transaction (x'20')

Bit 3 ONC/RPC Alias transaction (x'10')

Bit 4 WEB Alias transaction (x'08')

Bit 5 3270 Bridge transaction (x'04')

Bit 6 Reserved (x'02')

Bit 7 CICS BTS Run transaction

Byte 2
z/OS workload manager request (transaction).

Byte 3
Transaction definition information:

Bit 0 Taskdataloc = below (x'80')

Bit 1 Taskdatakey = cics (x'40')

Bit 2 Isolate = no (x'20')

Bit 3 Dynamic = yes (x'10')

Bits 4–7
Reserved

Byte 4
The type of the originating transaction:

X'01' None
X'02' Terminal
X'03' Transient data
X'04' START
X'05' Terminal-related START
X'06' CICS business transaction services (BTS) scheduler
X'07' Transaction manager domain (XM)-run transaction
X'08' 3270 bridge
X'09' Socket domain
X'0A' CICS web support (CWS)
X'0B' Internet Inter-ORB Protocol (IIOP)
X'0C' Resource Recovery Services (RRS)
X'0D' LU 6.1 session
X'0E' LU 6.2 (APPC) session
X'0F' MRO session
X'10' External Call Interface (ECI) session
X'11' IIOP domain request receiver
X'12' Request stream (RZ) instore transport
X'13' IP interconnectivity session
X'14' Event
X'15' JVM server

Byte 5
Transaction status information.

Byte 6
Transaction tracking origin data tag.

Byte 7
Recovery manager information:

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

Bit 2 Recovery manager - UOW resolved with indoubt action

Bit 3 Recovery manager - shunt

Bit 4 Recovery manager - unshunt

Bit 5 Recovery manager - indoubt failure

Bit 6 Recovery manager - resource owner failure

Bit 7 Reserved

371 (TYPE-C, 'OFCTYNME', 8 BYTES)
The facility name of the originating transaction. If the originating transaction is not associated with a facility, this field is null. The transaction facility type, if any, can be identified using byte 0 of the originating transaction flags, OTRANFLG (370), field.

372 (TYPE-C, 'OCLIPADR', 40 BYTES)
The IP address of the originating client or Telnet client.

373 (TYPE-C, 'PHNTWKID', 8 BYTES)
The network identifier of the CICS system of an immediately previous task in another CICS system with which this task is associated.

374 (TYPE-C, 'PHAPPLID', 8 BYTES)
The APPLID from previous hop data. This is the APPLID of the CICS system of a previous task in another CICS system with which this task is associated. See Previous hop data characteristics in Getting started for more information about previous hop data.

375 (TYPE-T, 'PHSTART', 8 BYTES)
The start time of the immediately previous task in another CICS system with which this task is associated.

376 (TYPE-P, 'PHTRANNO', 4 BYTES)
The task number of the immediately previous task in another CICS system with which this task is associated.

377 (TYPE-C, 'PHTRAN', 4 BYTES)
The transaction ID (TRANSID) of the immediately previous task in another CICS system with which this task is associated.

378 (TYPE-A, 'PHCOUNT', 4 BYTES)
The number of times there has been a request from one CICS system to another CICS system to initiate a task with which this task is associated.

402 (TYPE-A, 'EICTOTCT', 4 BYTES)
The total number of EXEC CICS commands issued by the user task.

405 (TYPE-A, 'TIASKTCT', 4 BYTES)
The number of EXEC CICS ASKTIME commands issued by the user task.

406 (TYPE-A, 'TITOTCT', 4 BYTES)
The total number of EXEC CICS ASKTIME, CONVERTTIME, and FORMATTIME commands issued by the user task.

408 (TYPE-A, 'BFDGSTCT', 4 BYTES)
The total number of EXEC CICS BIF DIGEST commands issued by the user task.

409 (TYPE-A, 'BFTOTCT', 4 BYTES)
The total number of EXEC CICS BIF DEEDIT and BIF DIGEST commands issued by the user task.

415 (TYPE-A, 'ECSIGECT', 4 BYTES)
The number of EXEC CICS SIGNAL EVENT commands issued by the user task.

416 (TYPE-A, 'ECEFOPCT', 4 BYTES)
The number of event filter operations performed by the user task.

417 (TYPE-A, 'ECEVNTCT', 4 BYTES)
The number of events captured by the user task.

418 (TYPE-A, 'ECSEVCCT', 4 BYTES)
The number of synchronous emission events captured by the user task.

449 (TYPE-A, 'MPPRTXCD', 4 BYTES)
Number of policy rule thresholds that this task has exceeded. This field is all nulls (0x00 bytes) if no thresholds have been exceeded or if the task has no policy rules applied to it.

464 (TYPE-A, 'NCGETCT', 4 BYTES)
The total number of requests to a named counter server to satisfy EXEC CICS GET COUNTER and EXEC CICS GET DCOUNTER commands issued by the user task.

Performance data in group DFHDATA

Descriptions of the performance data fields in the DFHDATA group, including the numeric identifier, type, and size of each field.

For more information about the time measurements used in some fields in this group, see "Clocks and time stamps" on page 335.

For more information about the elapsed time spent waiting for I/O operations and the relationship of that time to other time periods recorded for the transaction, see "Transaction wait (suspend) times" on page 339.

179 (TYPE-A, 'IMSREQCT', 4 BYTES)
The number of IMS (DBCTL) requests issued by the user task.

180 (TYPE-A, 'DB2REQCT', 4 BYTES)
The total number of DB2 EXEC SQL and Instrumentation Facility Interface (IFI) requests issued by the user task.

186 (TYPE-S, 'IMSWAIT', 12 BYTES)
The elapsed time during which the user task waited for DBCTL to service the IMS requests issued by the user task.

This field value is zero if IMS supports the open transaction environment (OTE).

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

187 (TYPE-S, 'DB2RDYQW', 12 BYTES)
The elapsed time during which the user task waited for a DB2 thread to become available.

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

188 (TYPE-S, 'DB2CONWT', 12 BYTES)
The elapsed time during which the user task waited for a DB2 connection to become available for use with the user task's open TCB.

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

395 (TYPE-A, 'WMQREQCT', 4 BYTES)
The total number of WebSphere MQ requests issued by the user task.

396 (TYPE-S, 'WMQGETWT', 12 BYTES)
The elapsed time during which the user task waited for WebSphere MQ to service the user task's GETWAIT request.

397 (TYPE-S, 'WMQASRBT', 12 BYTES)
The WebSphere MQ SRB time this transaction spent processing WebSphere MQ API requests. Add this field to the transaction CPU time field (USRCPUT) when considering the measurement of the total processor time consumed by a transaction. This field is zero for point-to-point messaging activity, but it is nonzero where WebSphere MQ API requests result in publish and subscribe type messaging.

Note: WebSphere MQ only returns this value to CICS when Class 3 accounting information is being collected in WebSphere MQ; if this information is not being collected, the field is always zero. To start collecting Class 3 accounting information, issue the command `START TRACE(ACCTG) DEST(SMF) CLASS(3)` in WebSphere MQ.

Performance data in group DFHDEST

Descriptions of the performance data fields in the DFHDEST group, including the numeric identifier, type, and size of each field.

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', 12 BYTES)
Elapsed time in which the user waited for VSAM transient data I/O. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

403 (TYPE-S, 'TDILWTT', 12 BYTES)
The elapsed time for which the user task waited for an intrapartition transient data lock (TDIPLOCK). For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. For more information about tasks suspended on resource type TDIPLOCK, see Resource type TDIPLOCK: waits for transient data intrapartition requests in Troubleshooting.

This field is a component of the task suspend time, SUSPTIME (014), field.

404 (TYPE-S, 'TDELWTT', 12 BYTES)
The elapsed time for which the user task waited for an extrapartition transient data lock (TDEPLOCK). For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. For more information about tasks suspended on resource type TDEPLOCK, see Resource type TDEPLOCK: waits for transient data extrapartition requests in Troubleshooting.

This field is a component of the task suspend time, SUSPTIME (014), field.

Performance data in group DFHDOCH

Descriptions of the performance data fields in the DFHDOCH group, including the numeric identifier, type, and size of each field.

223 (TYPE-A, 'DHDELCT', 4 BYTES)
The number of document handler DELETE requests issued by the user task.

226 (TYPE-A, 'DHCRECT', 4 BYTES)
The number of document handler CREATE requests issued by the user task.

227 (TYPE-A, 'DHINSCT', 4 BYTES)
The number of document handler INSERT requests issued by the user task.

228 (TYPE-A, 'DHSETCT', 4 BYTES)
The number of document handler SET requests issued by the user task.

229 (TYPE-A, 'DHRETCT', 4 BYTES)
The number of document handler RETRIEVE requests issued by the user task.

230 (TYPE-A, 'DHTOTCT', 4 BYTES)
The total number of document handler requests issued by the user task.

240 (TYPE-A, 'DHTOTDCL', 4 BYTES)
The total length of all documents created by the user task.

Performance data in group DFHFEPI

Descriptions of the performance data fields in the DFHFEPI group, including the numeric identifier, type, and size of each field.

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', 12 BYTES)
Elapsed time in which the user task waited for all FEPI services. For more information, see “Clocks and time stamps” on page 335, and “Transaction wait (suspend) times” on page 339.

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

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

Descriptions of the performance data fields in the DFHFILE group, including the numeric identifier, type, and size of each field.

For a breakdown by individual file of some of the information provided in group DFHFILE, you can request transaction resource monitoring. See “Transaction resource class data: Listing of data fields” on page 398 for details.

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', 12 BYTES)
Elapsed time in which the user task waited for file I/O. For more information, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.

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

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 is shown in Table 30.

Table 30. 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

Table 30. EXEC CICS file commands related to file control monitoring fields (continued)

EXEC CICS command	Monitoring fields
READNEXT	FCBRWCT and FCTOTCT
READNEXT UPDATE	FCBRWCT and FCTOTCT
READPREV	FCBRWCT and FCTOTCT
READPREV UPDATE	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.

174 (TYPE-S, 'RLSWAIT', 12 BYTES)
Elapsed time in which the user task waited for RLS file I/O. For more information, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.

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

175 (TYPE-S, 'RLSCPUT', 12 BYTES)
For applications that are not running in Threadsafe mode:

The RLS File Request CPU (SRB) time field (RLSCPUT) is the SRB CPU time this transaction spent processing RLS file requests. This field should be added to the transaction CPU time field (USRCPUT) when considering the measurement of the total CPU time consumed by a transaction. Also, this field cannot be considered a subset of any other single CMF field (including RLSWAIT). This is because the RLS field requests execute asynchronously under an MVS SRB which can be running in parallel with the requesting transaction. It is also possible for the SRB to complete its processing before the requesting transaction waits for the RLS file request to complete.

For applications that are running in Threadsafe mode:
There is no RLSCPUT field for applications that are running in Threadsafe mode because the requests are completed on the same TCB on which the application is running . In this case the CPU time for the request is already accumulated in the USRCPUT field.

176 (TYPE-S, 'CFDTWAIT', 12 BYTES)
The elapsed time in which the user task waited for a data table access request to the Coupling Facility Data Table server to complete. For more information, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.

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

426 (TYPE-S, 'FCXCWTT', 12 BYTES)
The elapsed time in which the user task waited for exclusive control of a VSAM control interval. This field counts time spent waiting on resource type FCXCSUSP, FCXDSUSP, FCXCPROT, or FCXDPROT. For more information, see

Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.

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

427 (TYPE-S, 'FCVSWTT', 12 BYTES)
The elapsed time in which the user task waited for a VSAM string. This field counts time spent waiting on resource type FCPSSUSP or FCSRSUSP. For more information, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.

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

Performance data in group DFHJOUR

Descriptions of the performance data fields in the DFHJOUR group, including the numeric identifer, type, and size of each field.

010 (TYPE-S, 'JCIOWTT', 12 BYTES)
Elapsed time for which the user task waited for journal (logstream) I/O. For more information, see "Clocks and time stamps" on page 335, and "Transaction wait (suspend) times" on page 339.

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

058 (TYPE-A, 'JNLWRTCT', 4 BYTES)
Number of journal write requests issued by the user task.

172 (TYPE-A, 'LOGWRTCT', 4 BYTES)
Number of CICS log stream write requests issued by the user task.

Performance data in group DFHMAPP

Descriptions of the performance data fields in the DFHMAPP group, including the numeric identifer, type, and size of each field.

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 BMS RECEIVE MAP, RECEIVE MAP FROM, SEND MAP, SEND TEXT, and SEND CONTROL requests issued by the user task.

Performance data in group DFHPROG

Descriptions of the performance data fields in the DFHPROG group, including the numeric identifer, type, and size of each field.

055 (TYPE-A, 'PCLINKCT', 4 BYTES)
Number of program LINK and INVOKE APPLICATION requests issued by the user task, including the link to the first program of the user task. This field does not include program LINK URM (user-replaceable module) requests.

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 called at transaction attach-time.

Note these points about remote transactions:

- If the CICS definition of the remote transaction does not specify a program name, this field contains blanks.
- If the CICS definition of the remote transaction specifies a program name, this field contains the name of the specified program. (This program 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 alternative program name, this field contains the name of the alternative 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.

For web service applications, this field contains the target application program name.

For a web alias transaction, this field contains the initial application program name called by the alias transaction. Web alias transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

For an ONC RPC transaction, this field contains the initial application program name called by the alias transaction. ONC RPC transactions can be identified using byte 1 of the transaction flags, TRANFLAG (164), field.

For an ECI over TCP/IP transaction, this field contains the name of the application program specified in the External Call Interface (ECI) request from the client application.

072 (TYPE-A, 'PCLURMCT', 4 BYTES)
Number of program LINK URM (user-replaceable module) requests issued by, or on behalf of, the user task.

A user-replaceable module (or user-replaceable program) is a CICS-supplied program that is always called at a particular point in CICS processing, as if it were part of the CICS code. You can modify the supplied program by including your own logic, or replace it with a version that you write yourself.

The user-replaceable programs are described in Customizing with user-replaceable programs in Developing system programs.

073 (TYPE-A, 'PCDPLCT', 4 BYTES)
Number of distributed program link (DPL) requests issued by the user task.

For a breakdown by program name and system identifier (sysid) of the individual distributed program link (DPL) requests, you can request transaction resource monitoring. For more details, see "Transaction resource class data: Listing of data fields" on page 398.

113 (TYPE-C, 'ABCODEO', 4 BYTES)
Original abend code. This field might not always be updated - see "Transactions that might not always update the ABCODEO and ABCODEC performance data fields" on page 365.

114 (TYPE-C, 'ABCODEC', 4 BYTES)
Current abend code. This field might not always be updated - see "Transactions that might not always update the ABCODEO and ABCODEC performance data fields" on page 365.

115 (TYPE-S, 'PCLOADTM', 12 BYTES)
Elapsed time in which the user task waited for fetches from DFHRPL or dynamic LIBRARY concatenations. Only fetches for programs with installed program definitions or autoinstalled as a result of application requests are included in this figure. However, installed programs 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 335, and "Program load time" on page 343.

286 (TYPE-A, 'PCDLCSDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the distributed program link (DPL) requests issued with the CHANNEL option by the user task. This total includes the length of any headers to the data.

287 (TYPE-A, 'PCDLCRDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all DPL RETURN CHANNEL commands issued by the user task. This total includes the length of any headers to the data.

306 (TYPE-A, 'PCLNKCCT', 4 BYTES)
Number of local program LINK and INVOKE APPLICATION requests, with the CHANNEL option, issued by the user task.

This field is a subset of the program LINK and INVOKE APPLICATION requests field, PCLINKCT (055).

307 (TYPE-A, 'PCXCLCCT', 4 BYTES)
Number of program XCTL requests issued with the CHANNEL option by the user task.

This field is a subset of the program XCTL requests field, PCXCTLCT (056).

308 (TYPE-A, 'PCDPLCCT', 4 BYTES)
Number of program distributed program link (DPL) requests issued with the CHANNEL option by the user task.

This field is a subset of the distributed program link requests field, PCDPLCT (073).

309 (TYPE-A, 'PCRTNCCT', 4 BYTES)
Number of remote pseudoconversational RETURN requests, with the CHANNEL option, issued by the user task.

310 (TYPE-A, 'PCRTNCDL', 4 BYTES)
The total length, in bytes, of the data in the containers of all the remote pseudoconversational RETURN CHANNEL commands issued by the user task. This total includes the length of any headers to the data.

Transactions that might not always update the ABCODEO and ABCODEC performance data fields

Due to the circumstances under which they are called, the following transactions might not always update the ABCODEO and ABCODEC performance data fields:

ASPF
ASPN
ASPO
ASPP
ASPQ
ASPR
ASP1
ASP2
ASP3
ASP7
ASP8

A transaction dump might not be taken either.

Performance data in group DFHRMI

Descriptions of the performance data fields in the DFHRMI group, including the numeric identifer, type, and size of each field.

Group DFHRMI is present in the performance class record only if RMI=YES is specified on the DFHMCT TYPE=INITIAL macro. For more information, see the RMI parameter on the DFHMCT TYPE=INITIAL macro in the *CICS Resource Definition Guide*.

001 (TYPE-S, 'RMITOTAL', 12 BYTES)
The total elapsed time spent in the CICS Resource Manager Interface (RMI).

For more information, see "Clocks and time stamps" on page 335, and "RMI elapsed and suspend time" on page 344.

002 (TYPE-S, 'RMIOTHER', 12 BYTES)
The total elapsed time spent in the CICS RMI for resource manager requests other than DB2, DBCTL, EXEC DLI, WebSphere MQ, CICSPlex SM, and CICS TCP/IP socket requests.

003 (TYPE-S, 'RMIDB2', 12 BYTES)
The total elapsed time spent in the CICS RMI for DB2 requests.

004 (TYPE-S, 'RMIDBCTL', 12 BYTES)
The total elapsed time spent in the CICS RMI for DBCTL requests.

005 (TYPE-S, 'RMIEXDLI', 12 BYTES)
The total elapsed time spent in the CICS RMI for EXEC DLI requests.

006 (TYPE-S, 'RMIMQM', 12 BYTES)
The total elapsed time spent in the CICS RMI for WebSphere MQ requests.

007 (TYPE-S, 'RMICPSM', 12 BYTES)
The total elapsed time spent in the CICS RMI for CICSPlex SM requests.

008 (TYPE-S, 'RMITCPIP', 12 BYTES)
The total elapsed time spent in the CICS RMI for CICS TCP/IP socket requests.

Performance data in group DFHSOCK

Descriptions of the performance data fields in the DFHSOCK group, including the numeric identifer, type, and size of each field.

241 (TYPE-S, 'SOIOWTT', 12 BYTES)
The elapsed time for which the user task waited for inbound socket I/O. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

242 (TYPE-A, 'SOBYENCT', 4 BYTES)
The number of bytes encrypted by the secure sockets layer for the user task.

243 (TYPE-A, 'SOBYDECT', 4 BYTES)
The number of bytes decrypted by the secure sockets layer for the user task.

245 (TYPE-C, 'TCPSRVCE', 8 BYTES)
The TCP/IP service name that attached the user task.

246 (TYPE-A, 'PORTNUM', 4 BYTES)
The TCP/IP port number of the TCP/IP service that attached the user task.

288 (TYPE-A, 'ISALLOCT, 4 BYTES)
The number of allocate session requests issued by the user task for sessions using IPIC.

289 (TYPE-A, 'SOEXTRCT', 4 BYTES)
The number of EXTRACT TCPIP and EXTRACT CERTIFICATE requests issued by the user task.

290 (TYPE-A,'SOCNPSCT',4 BYTES)
The total number of requests made by the user task to create a nonpersistent outbound socket.

291 (TYPE-A,'SOCPSCT',4 BYTES)
The total number of requests made by the user task to create a persistent outbound socket.

292 (TYPE-A,'SONPSHWM',4 BYTES)
The peak number of nonpersistent outbound sockets owned by the user task.

293 (TYPE-A,'SOPSHWM',4 BYTES)
The peak number of persistent outbound sockets owned by the user task.

294 (TYPE-A,'SORCVCT',4 BYTES)
The total number of receive requests issued for outbound sockets (persistent and nonpersistent) by the user task.

295 (TYPE-A,'SOCHRIN',4 BYTES)
The total number of bytes received on outbound sockets by the user task

296 (TYPE-A,'SOSENDCT',4 BYTES)
The total number of send requests issued for outbound sockets (persistent and nonpersistent) by the user task.

297 (TYPE-A,'SOCHROUT',4 BYTES)
The total number of bytes sent on outbound sockets by the user task.

298 (TYPE-A, 'SOTOTCT', 4 BYTES)
The total number of socket requests issued by the user task.

299 (TYPE-S, 'SOOIOWTT', 12 BYTES)
The total elapsed time that the user task waited on outbound sockets. For more

information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

300 (TYPE-S, 'ISIOWTT', 12 BYTES)
The elapsed time for which a user task waited for control at this end of an IPIC connection.

301 (TYPE-A, 'SOMSGIN1', 4 BYTES)
The number of inbound socket receive requests issued by the user task.

302 (TYPE-A, 'SOCHRIN1', 4 BYTES)
The number of characters received by inbound socket receive requests issued by the user task.

303 (TYPE-A, 'SOMSGOU1', 4 BYTES)
The number of inbound socket send requests issued by the user task.

304 (TYPE-A, 'SOCHROU1', 4 BYTES)
The number of characters sent by inbound socket send requests issued by the user task.

305 (TYPE-C, 'ISIPICNM', 8 BYTES)
The name of the IPIC connection for the TCP/IP service that attached the user task.

318 (TYPE-C, 'CLIPADDR', 40 BYTES)
The IP address of the client or Telnet client.

319 (TYPE-S, 'ISALWTT', 12 BYTES)
The elapsed time for which a user task waited for an allocate request for an IPIC session. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

320 (TYPE-A, 'SOCIPHER', 4 BYTES)
Identifies the code for the cipher suite that was selected during the SSL handshake for use on the inbound connection, for example X'0000002F'. For a list of the cipher suites that are supported by CICS and z/OS and their codes, see Cipher suites in Securing.

330 (TYPE-A, 'CLIPPORT', 4 BYTES)
The port number of the client or Telnet client.

Performance data in group DFHSTOR

Descriptions of the performance data fields in the DFHSTOR group, including the numeric identifer, type, and size of each field.

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 16 MB 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 for storage below the 16 MB line, in the UDSA.

095 (TYPE-A, 'SCUSRSTG', 8 BYTES)
Storage occupancy of the user task below the 16 MB 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 "Storage occupancy counts" on page 345.

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

107 (TYPE-A, 'SCUSRSTG', 8 BYTES)
Storage occupancy of the user task above the 16 MB line, in the EUDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see "Storage occupancy counts" on page 345.

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

118 (TYPE-A, 'SC24COCC', 8 BYTES)
Storage occupancy of the user task below the 16 MB line, in the CDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see "Storage occupancy counts" on page 345.

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

121 (TYPE-A, 'SC31COCC', 8 BYTES)
Storage occupancy of the user task above the 16 MB line, in the ECDSA. This measures the area under the curve of storage in use against elapsed time. For more information, see "Storage occupancy counts" on page 345.

441 (TYPE-A, 'SC64CGCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the bar, in the CICS dynamic storage area (GCDSA).

442 (TYPE-A, 'SC64CHWM', 4 BYTES)
Maximum amount (high-water mark) of user storage, rounded up to the next 4K, allocated to the user task above the bar, in the CICS dynamic storage area (GCDSA).

443 (TYPE-A, 'SC64UGCT', 4 BYTES)
Number of user-storage GETMAIN requests issued by the user task for storage above the bar, in the user dynamic storage area (GUDSA).

444 (TYPE-A, 'SC64UHWM', 4 BYTES)
Maximum amount (high-water mark) of user storage, rounded up to the next 4K, allocated to the user task above the bar, in the user dynamic storage area (GUDSA).

Table 31. User storage field ID by DSA (CICS key and user key)

Field	UDSA	EUDSA	CDSA	ECDSA	GCDSA	GUDSA
GETMAIN request count	054	105	117	120	441	443
High-water mark	033	106	116	119	442	444
Occupancy	095	107	118	121	n/a	n/a

Shared storage fields in group DFHSTOR

144 (TYPE-A, 'SC24SGCT', 4 BYTES)
Number of storage GETMAIN requests issued by the user task for shared storage below the 16 MB line, in the CDSA or SDSA.

145 (TYPE-A, 'SC24GSHR', 4 BYTES)
Number of bytes of shared storage obtained by the user task by using a GETMAIN request below the 16 MB line, in the CDSA or SDSA.

146 (TYPE-A, 'SC24FSHR', 4 BYTES)
Number of bytes of shared storage released by the user task by using a FREEMAIN request below the 16 MB line, in the CDSA or SDSA.

147 (TYPE-A, 'SC31SGCT', 4 BYTES)
Number of storage GETMAIN requests issued by the user task for shared storage above the 16 MB line, in the ECDSA or ESDSA.

148 (TYPE-A, 'SC31GSHR', 4 BYTES)
Number of bytes of shared storage obtained by the user task by using a GETMAIN request above the 16 MB line, in the ECDSA or ESDSA.

149 (TYPE-A, 'SC31FSHR', 4 BYTES)
Number of bytes of shared storage released by the user task by using a FREEMAIN request above the 16 MB line, in the ECDSA or ESDSA.

445 (TYPE-A, 'SC64SGCT', 4 BYTES)
Number of storage GETMAIN requests issued by the user task for shared storage above the bar, in the GCDSA or GSDSA.

446 (TYPE-A, 'SC64GSHR', 4 BYTES)
Amount of shared storage obtained by the user task by using a GETMAIN request above the bar, in the GCDSA or GSDSA. The total number of bytes obtained is rounded up to the next 4096 bytes, and the resulting number of 4K pages is displayed.

447 (TYPE-A, 'SC64FSHR', 4 BYTES)
Amount of shared storage released by the user task by using a FREEMAIN request above the bar, in the GCDSA or GSDSA. The total number of bytes obtained is rounded up to the next 4096 bytes, and the resulting number of 4K pages is displayed.

Table 32. Shared storage field ID by DSA

Field	CDSA or SDSA	ECDSA or ESDSA	GCDSA or GSDSA
GETMAIN request count	144	147	445
Shared storage obtained	145	148	446
Shared storage released	146	149	447

Program storage fields in group DFHSTOR

For more information on program storage see "Storage manager statistics" on page 681.

Note: If a task loads the same program several times, the fields in this group might not reflect the true high-water mark of program storage used by the task. The fields are incremented each time the LOAD command is issued, but if the program has already been loaded by the task, the existing copy of the program is used, meaning that only one copy of the program exists in storage. Because of this, for tasks that repeatedly load the same program, the data in the fields PCSTGHWM, PC24BHWM, PC31RHWM, PC31AHWM, PC31CHWM, PC24CHWM, PC24SHWM, PC31SHWM and PC24RHWM should be used with caution.

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 16 MB line.

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

122 (TYPE-A, 'PC31RHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16 MB 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 16 MB line. This field is a subset of PCSTGHWM (field id 087) that resides above the 16 MB line.

142 (TYPE-A, 'PC31CHWM', 4 BYTES)
Maximum amount (high-water mark) of program storage in use by the user task above the 16 MB 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 16 MB 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 16 MB 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 16 MB 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 16 MB 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

Descriptions of the performance data fields in the DFHSYNC group, including the numeric identifer, type, and size of each field.

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

Note:
1. A SYNCPOINT is implicitly issued as part of the task-detach processing.
2. A SYNCPOINT is issued at PSB termination for DBCTL.

173 (TYPE-S, 'SYNCTIME', 12 BYTES)
Total elapsed time for which the user task was dispatched and was processing syncpoint requests.

177 (TYPE-S, 'SRVSYWTT', 12 BYTES)
Total elapsed time in which the user task waited for syncpoint or resynchronization processing using the Coupling Facility data tables server to complete.

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

196 (TYPE-S, 'SYNCDLY', 12 BYTES)
The elapsed time in which the user task waited for a syncpoint request to be issued by its parent transaction. The user task was executing as a result of the parent task issuing a CICS BTS run-process or run-activity request to execute a process or activity synchronously. For more information, see “Clocks and time stamps” on page 335, and “Transaction wait (suspend) times” on page 339.

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

199 (TYPE-S, 'OTSINDWT', 12 BYTES)
The elapsed time in which the user task was dispatched or suspended indoubt (or both) while processing a syncpoint for an Object Transaction Service (OTS) syncpoint request. For more information, see “Clocks and time stamps” on page 335, and “Transaction wait (suspend) times” on page 339.

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

Performance data in group DFHTASK

Performance data fields in the DFHTASK group are described, including the numeric identifier, type, and size of each field.

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

004 (TYPE-C,'TTYPE',4 BYTES)
Transaction start type. The high-order bytes (0 and 1) are set as follows:

'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', 12 BYTES)
Total elapsed time during which the user task was dispatched on each CICS TCB under which the task ran. The TCB modes managed by the CICS dispatcher are: QR, RO, CO, FO, SZ ,RP, SL, SP, SO, EP, L8, L9, S8, TP, T8, X8, X9, and D2. Be aware that, for each CICS release, new TCB modes might be added to this list, or obsolete TCB modes might be removed. For more information about dispatch time and CPU time, see Transaction dispatch time and CPU time in Reference -> Monitoring.

008 (TYPE-S, 'USRCPUT', 12 BYTES)
Processor time for which the user task was dispatched on each CICS TCB under which the task ran. The TCB modes managed by the CICS dispatcher are: QR, RO, CO, FO, SZ, RP, SL, SP, SO, EP, L8, L9, S8, TP, T8, X8, X9, and D2. Be aware that, for each CICS release, new TCB modes might be added to this list, or obsolete TCB modes might be removed. For more information about dispatch time and CPU time, see Transaction dispatch time and CPU time in Reference -> Monitoring.

014 (TYPE-S, 'SUSPTIME', 12 BYTES)
Total elapsed wait time for which the user task was suspended by the dispatcher. This wait time includes these values:

- The elapsed time waiting for the first dispatch. This elapsed time also includes any delay incurred because of the limits set for the class of the transaction (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 Transaction wait (suspend) times in Reference -> Monitoring.

031 (TYPE-P, 'TRANNUM', 4 BYTES)
Transaction identification number. The transaction number field is normally a 4-byte packed decimal number. However, some CICS system tasks are identified by transaction numbers that comprise special characters, as follows:

' III' System initialization task

' TCP' Terminal control

These special identifiers are placed in bytes 2 - 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 32 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

065 (TYPE-A, 'ICSTACCT', 4 BYTES)
Total number of local interval control START requests, with the CHANNEL option, issued by the user task.

066 (TYPE-A, 'ICTOTCT', 4 BYTES)
Total number of Interval Control Start, Cancel, Delay, and Retrieve requests issued by the user task.

082 (TYPE-C, 'TRNGRPID', 28 BYTES)
The transaction group ID is assigned at transaction attach time, and can be used to correlate the transactions that CICS runs for the same incoming work request; for example, the CWXN and CWBA transactions for Web requests. This transaction group ID relationship is useful when applied to the requests that originate through the CICS Web, ECI over TCP/IP, or 3270 bridge interface, as indicated by the transaction origin in byte 4 of the transaction flags field (group name DFHTASK, field ID 164). For more information on using the transaction group ID, see Transaction tracking in Getting started in the CICS Intercommunication Guide.

097 (TYPE-C, 'NETUOWPX', 20 BYTES)
Fully qualified name by which the originating system is known to the z/OS Communications Server 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 z/OS Communications Server across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is non-z/OS Communications Server, 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-z/OS Communications Server terminal originators.

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 these fields:

- 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, 'NETUOWSX', 8 BYTES)
Name by which the network unit of work ID is known in 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 network unit of work ID passed as part of an ISC (APPC) or IRC (MRO) attach header.

The first 6 bytes of this field are a binary value derived from the system clock of the originating system and which can wrap round at intervals of several months.

The last 2 bytes of this field are for the period count. These bytes can change during the life of the task as a result of sync point activity.

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

102 (TYPE-S, 'DISPWTT', 12 BYTES)
Elapsed time for which the user task waited for redispatch. This time is the aggregate of the wait times between each event completion and user-task redispatch.

This field does not include the elapsed time spent waiting for first dispatch. This field is a component 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).

123 (TYPE-S, 'GNQDELAY', 12 BYTES)
The elapsed time waiting for a CICS task control global enqueue. For more information, see Clocks and time stamps in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

124 (TYPE-C, 'BRDGTRAN', 4 BYTES)
Bridge listener transaction identifier. For CICS 3270 Bridge transactions, this field is the name of the Bridge listener transaction that attached the user task.

125 (TYPE-S, 'DSPDELAY', 12 BYTES)
The elapsed time waiting for first dispatch.

This field is a component of the task suspend time, SUSPTIME (014), field. For more information, see Clocks and time stamps in Reference -> Monitoring.

126 (TYPE-S, 'TCLDELAY', 12 BYTES)
The elapsed time waiting for first dispatch, which was delayed because of the limits set for the transaction class of this transaction, TCLSNAME (166), being reached. For more information, see Clocks and time stamps in Reference -> Monitoring. This field is a component of the first dispatch delay, DSPDELAY (125), field.

127 (TYPE-S, 'MXTDELAY', 12 BYTES)
The elapsed time waiting for the first dispatch, which was delayed because of the limits set by the system parameter, MXT, being reached. The field is a component of the first dispatch delay, DSPDELAY (125), field.

128 (TYPE-S, 'LMDELAY', 12 BYTES)
The elapsed time that the user task waited to acquire a lock on a resource. A user task cannot explicitly acquire a lock on a resource, but many CICS modules lock resources on behalf of user tasks using the CICS lock manager (LM) domain.

For more information about CICS lock manager, see Investigating lock manager waits in Troubleshooting.

For information about times, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.This field is a component of the task suspend time, SUSPTIME (014), field.

129 (TYPE-S, 'ENQDELAY', 12 BYTES)
The elapsed time waiting for a CICS task control local enqueue. For more information, see Clocks and time stamps in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

132 (TYPE-T, 'RMUOWID', 8 BYTES)
The identifier of the unit of work (unit of recovery) for this task. Unit of recovery values are used to synchronize recovery operations among CICS and other resource managers, such as IMS and DB2.

163 (TYPE-C, 'FCTYNAME', 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified using byte 0 of the transaction flags, **TRANFLAG**, (164) field.

164 (TYPE-A, 'TRANFLAG', 8 BYTES)
Transaction flags, a string of 64 bits used for signaling transaction definition and status information:

Byte 0 The facility-type of the originating transaction:

Bit 0 Transaction facility name = none (x'80')

Bit 1 Transaction facility name = terminal (x'40')

If this bit is set, FCTYNAME and TERM contain the same terminal ID.

Bit 2 Transaction facility name = surrogate (x'20')

Bit 3 Transaction facility name = destination (x'10')

Bit 4 Transaction facility name = 3270 bridge (x'08')

Bits 5–7
Reserved

Byte 1 Transaction identification information:

Bit 0 System transaction (x'80')

Bit 1 Mirror transaction (x'40')

Bit 2 DPL mirror transaction (x'20')

Bit 3 ONC/RPC Alias transaction (x'10')

Bit 4 WEB Alias transaction (x'08')

Bit 5 3270 Bridge transaction (x'04')

Bit 6 Reserved (x'02')

Bit 7 CICS BTS Run transaction

Byte 2 z/OS workload manager request (transaction) completion information:

Bit 0 Report the total response time (begin-to-end phase) for completed work request (transaction).

Bit 1 Notify that the entire execution phase of the work request is complete.

Bit 2 Notify that a subset of the execution phase of the work request is complete.

Bit 3 This transaction has been reported to the z/OS workload manager as completing abnormally because it has tried to access DB2 and a “connection unavailable” response has been returned. This abnormal completion occurs when all the following are true:

- Bit 0 is set.
- CICS is not connected to DB2.
- The CICS-DB2 adapter is in standby mode (STANDBYMODE(RECONNECT) or STANDBYMODE(CONNECT)).
- CONNECTERROR(SQLCODE) is specified, causing the application to receive a -923 SQL code.

Bits 4-7
Reserved

Byte 3 Transaction definition information:

Bit 0 Taskdataloc = below (x'80')

Bit 1 Taskdatakey = cics (x'40')

Bit 2 Isolate = no (x'20')

Bit 3 Dynamic = yes (x'10')

Bits 4–7
Reserved

Byte 4 Transaction origin type:

X'01' None

X'02' Terminal

X'03' Transient data

X'04' START

X'05' Terminal-related START

X'06' CICS business transaction services (BTS) scheduler

X'07' Transaction manager domain (XM)-run transaction

X'08' 3270 bridge

X'09' Sockets domain

X'0A' CICS Web support (CWS)

X'0B' Internet Inter-ORB Protocol (IIOP)

X'0C' Resource Recovery Services (RRS)

X'0D' LU 6.1 session

X'0E' LU 6.2 (APPC) session

X'0F' MRO session

X'10' External Call Interface (ECI) session

X'11' IIOP domain request receiver

X'12' Request stream (RZ) instore transport

X'13' IPIC session

X'14' Event

X'15' JVMSERVER

Byte 5 Transaction status information:

Bit 0 The transaction origin

Bit 1 Reserved

Bit 2 Resource class record, or records, for this task

Bit 3 Identity class record, or records, for this task

Bit 4 Reserved

Bit 5 Reserved

Bit 6 Task purged on an open TCB

Note: If bit 6 is set, the task was purged while running on an open TCB, and its transaction timing clocks were left in an unreliable state. Because of this, the clocks are set to zero when the record is written by the CICS Monitoring Facility (CMF).

Bit 7 Task abnormally terminated

Byte 6 Transaction tracking origin data tag. For more information about how to set the transaction tracking origin data tag, see The SET_TRACKING_DATA call.

Note: The transaction tracking origin data tag can be set in the record only when it is the transaction origin. The transaction origin can be determined when bit 0 is set in the transaction status information byte 5 of the **TRANFLAG** field.

Byte 7 Recovery manager information:

Bit 0 Indoubt wait = no

Bit 1 Indoubt action = commit

Bit 2 Recovery manager, UOW resolved with indoubt action

Bit 3 Recovery manager, Shunt

Bit 4 Recovery manager, Unshunt

Bit 5 Recovery manager, Indoubt failure

Bit 6 Recovery manager, Resource owner failure

Bit 7 Reserved

Note: Bits 2 through 6 are reset on a SYNCPOINT request when the MNSYNC=YES option is specified.

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', 12 BYTES)
The total elapsed time spent in the CICS Resource Manager Interface (RMI). For more information, see Clocks and time stamps in Reference -> Monitoring,

Transaction wait (suspend) times in Reference -> Monitoring, and RMI elapsed and suspend time in Reference -> Monitoring.

171 (TYPE-S, 'RMISUSP', 12 BYTES)
The total elapsed time that the task was suspended by the CICS dispatcher while in the CICS Resource Manager Interface (RMI). For more information, see Clocks and time stamps in Reference -> Monitoring, Transaction wait (suspend) times in Reference -> Monitoring, and RMI elapsed and suspend time in Reference -> Monitoring. The field is a component of the task suspend time, SUSPTIME (014), field and also the RMITIME (170) field.

181 (TYPE-S, 'WTEXWAIT', 12 BYTES)
The elapsed time that the user task waited for one or more ECBs, passed to CICS by the user task using the **EXEC CICS WAIT EXTERNAL ECBLIST** command, to be posted by the MVS POST command. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

182 (TYPE-S, 'WTCEWAIT', 12 BYTES)
The elapsed time that the user task waited for one of these events:

- One or more ECBs, passed to CICS by the user task using the **EXEC CICS WAITCICS ECBLIST** command, to be posted by the MVS POST command. The user task can wait on one or more ECBs. If it waits on more than one, it is dispatchable as soon as one of the ECBs is posted.
- Completion of an event initiated by the same or by another user task. The event is usually be the posting, at the expiration time, of a timer-event control area provided in response to an **EXEC CICS POST** command. The **EXEC CICS WAIT EVENT** command provides a method of directly giving up control to some other task until the event being waited on is completed.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

183 (TYPE-S, 'ICDELAY', 12 BYTES)
The elapsed time that the user task waited as a result of issuing one of the following commands:

- An interval control **EXEC CICS DELAY** command for a specified time interval.
- An interval control **EXEC CICS DELAY** command for a specified time of day to expire.
- An interval control **EXEC CICS RETRIEVE** command with the WAIT option specified.

For more information, see Clocks and time stamps in Reference -> Monitoring, and Transaction wait (suspend) times in Reference -> Monitoring.This field is a component of the task suspend time, SUSPTIME (014), field.

184 (TYPE-S, 'GVUPWAIT', 12 BYTES)
The elapsed time that the user task waited as a result of giving up control to another task. A user task can give up control in many ways. Some examples are application programs that use one or more of the following EXEC CICS API or SPI commands:

- The **EXEC CICS SUSPEND** command. This command causes the issuing task to give upcontrol to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- The **EXEC CICS CHANGE TASK PRIORITY** command. This command immediately changes the priority of the issuing task and causes the task to give up control for it to be dispatched at its new priority. The task is not redispatched until tasks of higher or equal priority, and that are also dispatchable, have been dispatched.
- The **EXEC CICS DELAY** command with INTERVAL (0). This command causes the issuing task to give up control to another task of higher or equal dispatching priority. Control is returned to this task as soon as no other task of a higher or equal priority is ready to be dispatched.
- The **EXEC CICS POST** command requesting notification that a specified time has expired. This command causes the issuing task to give up control so that CICS has the opportunity to post the time-event control area.
- The **EXEC CICS PERFORM RESETTIME** command to synchronize the CICS date and time with the MVS system date and time of day.
- The **EXEC CICS START TRANSID** command with the ATTACH option.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

190 (TYPE-C, 'RRMSURID', 16 BYTES)
RRMS/MVS unit-of-recovery ID (URID).

191 (TYPE-S, 'RRMSWAIT', 12 BYTES)
The elapsed time in which the user task waited indoubt using resource recovery services for EXCI.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

192 (TYPE-S, 'RQRWAIT', 12 BYTES)
The elapsed time during which the request receiver user task CIRR (or user specified transaction ID) waited for any outstanding replies to be satisfied.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

193 (TYPE-S, 'RQPWAIT', 12 BYTES)
The elapsed time during which the request processor user task CIRP waited for any outstanding replies to be satisfied.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

194 (TYPE-C, 'OTSTID', 128 BYTES)
This field is the first 128 bytes of the Object Transaction Service (OTS) Transaction ID (TID).

195 (TYPE-S, 'RUNTRWTT', 12 BYTES)
The elapsed time in which the user task waited for completion of a transaction that ran as a result of the user task issuing a CICS BTS run process request and a run activity request synchronously.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

247 (TYPE-S, 'DSCHMDLY', 12 BYTES)
The elapsed time in which the user task waited for redispatch after a CICS Dispatcher change-TCB mode request was issued by or on behalf of the user task. For example, a change-TCB mode request from a CICS L8 or S8 mode TCB back to the CICS QR mode TCB might have to wait for the QR TCB because another task is currently dispatched on the QR TCB. This field is a component of the task suspend time, SUSPTIME (014), field.

249 (TYPE-S, 'QRMODDLY', 12 BYTES)
The elapsed time for which the user task waited for redispatch on the CICS QR mode TCB. This time is the aggregate of the wait times between each event completion and user-task redispatch. This field does not include the elapsed time spent waiting for the first dispatch. The QRMODDLY field is a component of the task suspend time, SUSPTIME (014), field, and also the redispatch wait, DISPWTT (102), field.

250 (TYPE-S, 'MXTOTDLY', 12 BYTES)
The elapsed time in which the user task waited to obtain a CICS L8 or L9 mode open TCB, because the region had reached the limit set by CICS for these TCBs. L8 and L9 mode open TCBs are used by OPENAPI application programs or by task-related user exit programs that have been enabled with the OPENAPI option, for example, the CICS-DB2 adapter, when CICS connects to DB2 Version 6 or later and the CICS-MQ adapter, when CICS connects to Websphere MQ Version 6 or later.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.This field is a component of the task suspend time, SUSPTIME (014), field.

251 (TYPE-A, 'TCBATTCT', 4 BYTES)
The number of CICS TCBs attached by or on behalf of the user task.

252 (TYPE-A, 'DSTCBHWM', 4 BYTES)
The peak number of CICS open TCBs (in TCB modes L8, L9, S8, T8, X8, and X9) that have been concurrently allocated to the user task.

253 (TYPE-S, 'JVMTIME', 12 BYTES)
The total elapsed time spent in the JVM by the user task. For more information, see JVM elapsed time, suspend time, and cleanup time in Reference -> Monitoring.

254 (TYPE-S, 'JVMSUSP', 12 BYTES)
The elapsed time for which the user task was suspended by the CICS dispatcher while running in the JVM. For more information, see JVM elapsed time, suspend time, and cleanup time in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

255 (TYPE-S, 'QRDISPT', 12 BYTES)
The elapsed time for which the user task was dispatched on the CICS QR TCB. For more information, see Clocks and time stamps in Reference -> Monitoring.

256 (TYPE-S, 'QRCPUT', 12 BYTES)
The processor time for which the user task was dispatched on the CICS QR TCB. For more information, see Clocks and time stamps in Reference -> Monitoring.

257 (TYPE-S, 'MSDISPT', 12 BYTES)
Elapsed time for which the user task was dispatched on each CICS TCB. The CICS TCB modes are used as follows:

- RO and FO are always used.
- CO is used if **SUBTSKS=1** is specified as a system initialization parameter.
- SZ is used if FEPI is active.
- RP is used if ONC/RPC is installed and active.
- SL, SO, and SP are used if **TCPIP=YES** is specified as a system initialization parameter. Mode SL is used by the CICS support for TCP/IP (TCP/IP Service) Listener system transaction CSOL. Mode SO is used to process the CICS support for TCP/IP socket requests issued by or on behalf of the user task. Mode SP is the CICS support for TCP/IP sockets IPT task (Initial Pthread TCB) and also owns all the SSL pthreads (S8 TCBs).
- D2 is used to stop DB2 protected threads.
- EP is used for event processing.
- CICS creates a TP mode TCB for every JVMSERVER resource definition that is installed and enabled. The TP TCB owns the IPT task (Initial Process Thread TCB), the Language Environment enclave, the JVM, the THRD TCB pool, and the T8 TCBs for that JVM server.

For more information, see Clocks and time stamps in Reference -> Monitoring.

258 (TYPE-S, 'MSCPUT', 12 BYTES)
The processor time for which the user task was dispatched on each CICS TCB. The usage of each CICS TCB is shown in the description for field **MSDISPT** (field ID 257 in group DFHTASK). For more information, see Clocks and time stamps in Reference -> Monitoring.

259 (TYPE-S, 'L8CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS L8 mode TCB. When a transaction starts an OPENAPI application program defined with EXECKEY=CICS, or a task-related user exit program that has been enabled with the OPENAPI option, CICS allocates a CICS L8 mode TCB to the task. (An L8 mode TCB can also be allocated if the OPENAPI program is defined with EXECKEY=USER, but the storage protection facility is inactive.) After a task has been allocated an L8 mode TCB, that same TCB remains associated with the task until the transaction is detached. For more information on this field, see Clocks and time stamps in Reference -> Monitoring.

261 (TYPE-S, 'S8CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS S8 mode TCB. A transaction is allocated a CICS S8 mode TCB when it uses the secure sockets layer (SSL) during client certificate negotiation. The S8 mode TCB remains associated with the same task for the life of the SSL request. For more information, see Clocks and time stamps in Reference -> Monitoring.

262 (TYPE-S, 'KY8DISPT', 12 BYTES)
The total elapsed time during which the user task was dispatched by the CICS dispatcher on a CICS Key 8 mode TCB:

- An L8 mode TCB is allocated when a transaction calls an OPENAPI application program defined with EXECKEY=CICS or a task-related user exit program that has been enabled with the OPENAPI option. The TCB remains associated with the task until the transaction is detached.

- An S8 mode TCB is allocated when a transaction is using the secure sockets layer (SSL) during client certificate negotiation. The S8 mode TCB remains associated with the same task for the life of the SSL request.
- A T8 mode TCB is allocated when a transaction is using a JVM server to perform multithreaded processing. When a thread is allocated a T8 mode TCB, that same TCB remains associated with the thread until the processing completes.
- An X8 mode TCB is allocated when a transaction calls a C or C++ program that was compiled with the XPLINK option and that is defined with EXECKEY=CICS. The TCB remains associated with the task until the program ends.

This field is a component of the task dispatch time field, **USRDISPT** (field ID 007 in group DFHTASK).

263 (TYPE-S, 'KY8CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher on a CICS Key 8 mode TCB. The usage of the CICS Key 8 mode TCBs is shown in the description for field **KY8DISPT** (field ID 262 in group DFHTASK). This field is a component of the task CPU time field, **USRCPUT** (field ID 008 in group DFHTASK).

264 (TYPE-S, 'KY9DISPT', 12 BYTES)
The total elapsed time during which the user task was dispatched by the CICS dispatcher on a CICS Key 9 mode TCB:

- An L9 mode TCB is allocated when a transaction calls an OPENAPI application program defined with EXECKEY=USER. The TCB remains associated with the task until the transaction is detached.
- An X9 mode TCB is allocated when a transaction calls a C or C++ program that was compiled with the XPLINK option and that is defined with EXECKEY=USER. The TCB remains associated with the task until the program ends.

This field is a component of the task dispatch time field, USRDISPT (field ID 007 in group DFHTASK).

265 (TYPE-S, 'KY9CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher on a CICS Key 9 mode TCB. The usage of the CICS Key 9 mode TCBs is shown in the description for field **KY9DISPT** (field ID 264 in group DFHTASK). This field is a component of the task CPU time field, USRCPUT (field ID 008 in group DFHTASK).

266 (TYPE-S, 'L9CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS L9 mode TCB. When a transaction calls an OPENAPI application program that is defined with EXECKEY=USER it is allocated and uses a CICS L9 mode TCB. If the storage protection facility is inactive, an L8 mode TCB is used instead of an L9 mode TCB. When a task has been allocated an L9 mode TCB, that same TCB remains associated with the task until the transaction is detached. This field is a component of the total task CPU time field, USRCPUT (field ID 008 in group DFHTASK), and the task key 9 CPU time field, KY9CPUT (field ID 265 in group DFHTASK).

268 (TYPE-S, 'DSTCBMWT', 12 BYTES)
The elapsed time that the user task spent in TCB mismatch waits; that is, waiting because no available TCB matched the request, but at least one non matching TCB was free.

269 (TYPE-S, 'RODISPT', 12 BYTES)
The elapsed time during which the user task was dispatched by the CICS dispatcher on the CICS RO mode TCB. The RO TCB is used for loading programs, unless the command to load the program (EXEC CICS LOAD, XCTL, or LINK) is issued by an application that is currently running on an open TCB. In that situation, the open TCB is used to load the program instead of the RO TCB. The CICS RO mode TCB is also used for opening and closing CICS data sets, issuing RACF calls, and similar tasks. This field is a component of the task dispatch time field, USRDISPT (group name: DFHTASK, field ID: 007) and the task miscellaneous TCB dispatch time field, MSDISPT (group name: DFHTASK, field ID: 257).

270 (TYPE-S, 'ROCPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher on the CICS RO mode TCB. The RO TCB is used for loading programs, unless the command to load the program (EXEC CICS LOAD, XCTL, or LINK) is issued by an application that is currently running on an open TCB. In that situation, the open TCB is used to load the program instead of the RO TCB. The CICS RO mode TCB is also used for opening and closing CICS data sets, issuing RACF calls, and similar tasks. This field is a component of the task CPU time field, USRCPUT (group name: DFHTASK, field ID: 008) and the task miscellaneous TCB CPU time field, MSCPUT (group name: DFHTASK, field ID: 258).

271 (TYPE-S, 'X8CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS X8 mode TCB. When a transaction calls a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=CICS, it is allocated and uses a CICS X8 mode TCB. An X8 mode TCB can also be allocated if the program is defined with EXECKEY=USER, but the storage protection facility is inactive. After a task has been allocated an X8 mode TCB, that same TCB remains associated with the task until the program completes. This field is a component of the total task CPU time field, USRCPUT (field ID 008 in group DFHTASK), and the task key 8 CPU time field, KY8CPUT (field ID 263 in group DFHTASK).

272 (TYPE-S, 'X9CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS X9 mode TCB. When a transaction calls a C or C++ program that was compiled with the XPLINK option, and that is defined with EXECKEY=USER, it is allocated and uses a CICS X9 mode TCB. (If the storage protection facility is inactive, an X8 mode TCB is used instead of an X9 mode TCB.) After a task has been allocated an X9 mode TCB, that same TCB remains associated with the task until the program completes. This field is a component of the total task CPU time field, USRCPUT (field ID 008 in group DFHTASK), and the task key 9 CPU time field, KY9CPUT (field ID 265 in group DFHTASK).

273 (TYPE-S, 'JVMITIME', 12 BYTES)
The elapsed time spent initializing the JVM environment. For more information, see Clocks and time stamps in Reference -> Monitoring.

275 (TYPE-S, 'JVMRTIME', 12 BYTES)
Reserved field, returns zero.

279 (TYPE-S, 'DSMMSCWT', 12 BYTES)
The elapsed time that the user task spent waiting because no TCB was available and a TCB was not created because of MVS storage constraints. This field is a component of the task suspend time, SUSPTIME (014), field.

281 (TYPE-S, 'MAXSTDLY', 12 BYTES)
The elapsed time for which the user task waited to obtain a CICS SSL TCB (S8 mode), because the CICS system reached the limit set by the system initialization parameter MAXSSLTCBS. The S8 mode open TCBs are used exclusively by secure sockets layer (SSL) pthread requests issued by or on behalf of a user task. For more information, see Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

282 (TYPE-S, 'MAXXTDLY', 12 BYTES)
The elapsed time for which the user task waited to obtain a CICS XP TCB (X8 or X9 mode), because the CICS system reached the limit set by CICS for these types of TCB. The X8 and X9 mode open TCBs are used exclusively by C and C++ programs that were compiled with the XPLINK option. For more information, see Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

283 (TYPE-S, 'MAXTTDLY', 12 BYTES)
The elapsed time for which the user task waited to obtain a T8 TCB, because the CICS system reached the limit of available threads. The T8 mode open TCBs are used by a JVM server to perform multithreaded processing. Each T8 TCB runs under one thread. The thread limit is 2000 for each CICS region and each JVM server in a CICS region can have up to 256 threads. For more information, see Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

285 (TYPE-S, 'PTPWAIT', 12 BYTES)
The elapsed time for which the user task waited for the 3270 bridge partner transaction to complete. For more information, see Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

345 (TYPE-A, 'ICSTACDL', 4 BYTES)
Total length, in bytes, of the data in the containers of all the locally executed START CHANNEL requests issued by the user task. This total includes the length of any headers to the data.

346 (TYPE-A, 'ICSTRCCT', 4 BYTES)
Total number of interval control START CHANNEL requests, to be run on remote systems, issued by the user task.

347 (TYPE-A, 'ICSTRCDL', 4 BYTES)
Total length, in bytes, of the data in the containers of all the remotely executed START CHANNEL requests issued by the user task. This total includes the length of any headers to the data.

348 (TYPE-S, 'ROMODDLY', 12 BYTES)
The elapsed time for which the user task waited for redispatch on the CICS RO TCB. This time is the aggregate of the wait times between each event completion and user-task redispatch. The ROMODDLY field is a component of the task suspend time, SUSPTIME (014), field, and also the redispatch wait, DISPWTT (102), field.

349 (TYPE-S, 'SOMODDLY', 12 BYTES)
The elapsed time for which the user task waited for redispatch on the CICS SO TCB. This time is the aggregate of the wait times between each event completion and user-task redispatch. The SOMODDLY field is a component of the task suspend time, SUSPTIME (014), field, and also the redispatch wait, DISPWTT (102), field.

400 (TYPE-S, 'T8CPUT', 12 BYTES)
The processor time during which the user task was dispatched by the CICS dispatcher domain on a CICS T8 mode TCB. T8 mode TCBs are used by a JVM server to perform multithreaded processing. When a thread is allocated a T8 mode TCB, that same TCB remains associated with the thread until the processing completes. This field is a component of the total task CPU time field, USRCPUT (field ID 008 in group DFHTASK), and the task key 8 CPU time field, KY8CPUT (field ID 263 in group DFHTASK).

401 (TYPE-S, 'JVMTHDWT', 12 BYTES)
The elapsed time that the user task waited to obtain a JVM server thread because the CICS system had reached the thread limit for a JVM server in the CICS region. This field is a component of the task suspend time, SUSPTIME (014), field.

429 (TYPE-S, 'DSAPTHWT', 12 BYTES
The dispatcher allocated pthread wait time. This is the time that the transaction had to wait for a Liberty pthread to be allocated during links to Liberty programs. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring. This field is a component of the task suspend time, SUSPTIME (014), field.

430 (TYPE-C, 'CECMCHTP', 4 BYTES)
The CEC machine type, in EBCDIC, for the physical hardware environment where the CICS region is running. CEC (central electronics complex) is a commonly used synonym for CPC (central processing complex).

431 (TYPE-C, 'CECMDLID', 16 BYTES)
The CEC model number, in EBCDIC, for the physical hardware environment where the CICS region is running.

433 (TYPE-A, 'MAXTASKS', 4 BYTES)
The MXT or MAXTASKS value, expressed as a number of tasks, for the CICS region at the time the user task was attached.

434 (TYPE-A, 'CURTASKS', 4 BYTES)
The current number of active user transactions in the system at the time the user task was attached.

436 (TYPE-S, 'CPUTONCP', 12 BYTES)
The total task processor time on a standard processor for which the user task was dispatched on each CICS TCB under which the task ran.

This field is a component of the task CPU time field, USRCPUT (field ID 008 in group DFHTASK). To calculate the task processor time that was spent on a specialty processor (zIIP or zAAP), subtract the time recorded in the CPUTONCP field from the time recorded in the USRCPUT field.

Note: The times shown in the CPUTONCP and OFFLCPUT fields are only available when running on a system that supports the Extract CPU Time instruction service that is available on IBM z9® or later hardware. For z/OS, Version 1 Release 13, the PTF for APAR OA38409 must also be applied.

437 (TYPE-S, 'OFFLCPUT', 12 BYTES)
The total task processor time that was spent on a standard processor but was eligible for offload to a specialty processor (zIIP or zAAP).

This field is a component of the task CPU time field, USRCPUT (field ID 008 in group DFHTASK), and also a component of the standard CPU time field, CPUTONCP (field ID 436 in group DFHTASK). To calculate the task processor time spent on a standard processor that was not eligible for offload to a specialty processor, subtract the time recorded in the OFFLCPUT field from the time recorded in the CPUTONCP field.

Note: The times shown in the CPUTONCP and OFFLCPUT fields are only available when running on a system that supports the Extract CPU Time instruction service that is available on IBM z9 or later hardware. For z/OS, Version 1 Release 13, the PTF for APAR OA38409 must also be applied.

451 (TYPE-C, 'ACAPPLNM', 64 BYTES)
The 64-character name of the application in the application context data.

452 (TYPE-C, 'ACPLATNM', 64 BYTES)
The 64-character name of the platform in the application context data.

453 (TYPE-A, 'ACMAJVER', 4 BYTES)
The major version of the application in the application context data, expressed as a 4-byte binary value.

454 (TYPE-A, 'ACMINVER', 4 BYTES)
The minor version of the application in the application context data, expressed as a 4-byte binary value.

455 (TYPE-A, 'ACMICVER', 4 BYTES)
The micro version of the application in the application context data, expressed as a 4-byte binary value.

456 (TYPE-C, 'ACOPERNM', 64 BYTES)
The 64-character name of the operation in the application context data.

For a description of TCB modes, see TCB statistics in Improving performance.

Performance data in group DFHTEMP

Descriptions of the performance data fields in the DFHTEMP group, including the numeric identifer, type, and size of each field.

For a breakdown by individual temporary storage queue of the information provided in group DFHTEMP, you can request transaction resource monitoring. See “Transaction resource class data: Listing of data fields” on page 398 for details.

011 (TYPE-S, 'TSIOWTT', 12 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 335, and “Transaction wait (suspend) times” on page 339.

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

044 (TYPE-A, 'TSGETCT', 4 BYTES)
Number of temporary storage GET requests to auxiliary or main temporary storage 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 the temporary storage READQ (TSGETCT), READQ shared (TSGETSCT), WRITEQ AUX (TSPUTACT), WRITEQ MAIN (TSPUTMCT), WRITEQ shared (TSPUTSCT), and DELETEQ requests issued by the user task.

178 (TYPE-S, 'TSSHWAIT', 12 BYTES)
Elapsed time that the user task waited for an asynchronous shared temporary storage request to a temporary storage data server to complete. For more information, see "Clocks and time stamps" on page 335, and "Transaction wait (suspend) times" on page 339.

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

460 (TYPE-A, 'TSGETSCT', 4 BYTES)
Number of temporary storage GET requests from shared temporary storage issued by the user task.

461 (TYPE-A, 'TSPUTSCT', 4 BYTES)
Number of temporary storage PUT requests to shared temporary storage issued by the user task.

Performance data in group DFHTERM

Descriptions of the performance data fields in the DFHTERM group, including the numeric identifier, type, and size of each field.

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.

009 (TYPE-S, 'TCIOWTT', 12 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 in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

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

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

084 (TYPE-A, 'TCCHROU1', 4 BYTES)
Number of characters sent to the principal terminal facility of the task, 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', 12 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 in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

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

133 (TYPE-S, 'LU61WTT', 12 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 because of LUTYPE6.1 syncpoint flows. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

134 (TYPE-S, 'LU62WTT', 12 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 because of LUTYPE6.2 (APPC) syncpoint flows. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

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.

165 (TYPE-A, 'TERMINFO', 4 BYTES)
Terminal or session information for the principal facility of this task, as identified in the 'TERM' field id 002. This field is null if the task is not associated with a terminal or session facility.

Byte 0 Identifies whether this task is associated with a terminal or session. This field can be set to one of the following values:

X'00' None

X'01' Terminal

X'02' Session

Byte 1 If the principal facility for this task is a session (Byte 0 = x'02'), this field identifies the session type. This field can be set to one of the following values:

X'00' None

X'01' IRC

X'02' IRC XM

X'03' IRC XCF

X'04' LU61

X'05' LU62 Single

X'06' LU62 Parallel

Byte 2 Identifies the access method defined for the terminal ID or session ID in field TERM. This field can be set to one of the following values:

X'00' None

X'01' Communications Server

X'02' Reserved

X'03' BSAM

X'04' Reserved

X'05' Reserved

X'06' BGAM

X'07' CONSOLE

Byte 3 Identifies the terminal or session type for the terminal id or session id in TERM.

- See RDO Typeterm

For a list of the typeterm definitions, see ASSIGN TERMCODE in Reference -> Application development.

169 (TYPE-C, 'TERMCNNM', 4 BYTES)
Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (sysid).

A terminal facility can be identified as a session by using byte 0 of the terminal information, TERMINFO (165), field. If the value is x'02', the terminal facility is a session.

197 (TYPE-C, 'NETID', 8 BYTES)
NETID if a network qualified name has been received from the

Communications Server. If it is a resource and the network qualified name has not yet been received, NETID is 8 blanks. In all other cases, it is nulls.

198 (TYPE-C, 'RLUNAME', 8 BYTES)
Real network name if a network qualified name has been received from the Communications Server. In all other cases, this field is the same as LUNAME (field ID 111). For non-Communications Server resources, it is nulls.

343 (TYPE-S, 'TCALWTT', 12 BYTES)
The elapsed time for which a user task waited for an allocate request for an MRO (Inter-Region Communication), LU6.1, or LU6.2 session. For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

This field is a component of the task suspend time, SUSPTIME (014), field.

Performance data in group DFHWEBB

Descriptions of the performance data fields in the DFHWEBB group, including the numeric identifer, type, and size of each field.

224 (TYPE-A, 'WBREADCT', 4 BYTES)
The number of CICS web support READ HTTPHEADER, READ FORMFIELD, and READ QUERYPARM requests issued by the user task.

225 (TYPE-A, 'WBWRITCT', 4 BYTES)
The number of CICS web support WRITE HTTPHEADER requests issued by the user task.

231 (TYPE-A, 'WBRCVCT', 4 BYTES)
The number of CICS web support RECEIVE requests issued by the user task.

232 (TYPE-A, 'WBCHRIN', 4 BYTES)
The number of bytes received by the CICS web support RECEIVE requests issued by the user task.

233 (TYPE-A, 'WBSENDCT', 4 BYTES)
The number of CICS web support SEND requests issued by the user task.

234 (TYPE-A, 'WBCHROUT', 4 BYTES)
The number of bytes sent by the CICS web support SEND requests issued by the user task.

235 (TYPE-A, 'WBTOTWCT', 4 BYTES)
The total number of CICS web support requests issued by the user task.

236 (TYPE-A, 'WBREPRCT', 4 BYTES)
The number of reads from the repository in temporary storage issued by the user task.

237 (TYPE-A, 'WBREPWCT', 4 BYTES)
The number of writes to the repository in temporary storage issued by the user task.

238 (TYPE-A, 'WBEXTRCT', 4 BYTES)
The number of CICS web support EXTRACT requests issued by the user task.

239 (TYPE-A, 'WBBRWCT', 4 BYTES)
The number of CICS web support browsing requests for HTTPHEADER, FORMFIELD, and QUERYPARM (STARTBROWSE, READNEXT, and ENDBROWSE) issued by the user task.

331 (TYPE-A, 'WBREDOCT', 4 BYTES)
The number of CICS web support READ HTTPHEADER requests issued by the user task when CICS is an HTTP client.

332 (TYPE-A, 'WBWRTOCT', 4 BYTES)
The number of CICS web support WRITE HTTPHEADER requests issued by the user task when CICS is an HTTP client.

333 (TYPE-A, 'WBRCVIN1', 4 BYTES)
The number of CICS web support RECEIVE and CONVERSE requests issued by the user task when CICS is an HTTP client.

334 (TYPE-A, 'WBCHRIN1', 4 BYTES)
The number of bytes received by the CICS web support RECEIVE and CONVERSE requests issued by the user task when CICS is an HTTP client. This number includes the HTTP headers for the response.

335 (TYPE-A, 'WBSNDOU1', 4 BYTES)
The number of CICS web support SEND and CONVERSE requests issued by the user task when CICS is an HTTP client.

336 (TYPE-A, 'WBCHROU1', 4 BYTES)
The number of bytes sent by the CICS web support SEND and CONVERSE requests issued by the user task when CICS is an HTTP client. This number includes the HTTP headers for the request.

Note: When requests are made using the **WEB CONVERSE** command, the requests increment both the Send and Receive request counts (WBSNDOU1 and WBRCVIN1) and the counts of characters sent and received (WBCHRIN1 and WBCHROU1).

337 (TYPE-A, 'WBPARSCT', 4 BYTES)
The number of CICS web support PARSE URL requests issued by the user task.

338 (TYPE-A, 'WBBRWOCT', 4 BYTES)
The number of CICS web support BROWSE HTTPHEADER requests (STARTBROWSE, READNEXT, and ENDBROWSE) issued by the user task when CICS is an HTTP client.

340 (TYPE-A, 'WBIWBSCT', 4 BYTES)
The number of **EXEC CICS INVOKE SERVICE and EXEC CICS INVOKE WEBSERVICE** requests issued by the user task.

341 (TYPE-A, 'WBREPRDL', 4 BYTES)
The total length, in bytes, of the data read from the repository in temporary storage by the user task.

342 (TYPE-A, 'WBREPWDL', 4 BYTES)
The total length, in bytes, of the data written to the repository in temporary storage by the user task.

380 (TYPE-C, 'WBURIMNM', 8 BYTES)
For CICS web support, Atom feeds, and web service applications, the name of the URIMAP resource definition that was mapped to the URI of the inbound request that was processed by this task.

381 (TYPE-C, 'WBPIPLNM', 8 BYTES)
For web service applications, the name of the PIPELINE resource definition that was used to provide information about the message handlers that act on the service request processed by this task.

382 (TYPE-C, 'WBATMSNM', 8 BYTES)
For Atom feeds, the name of the ATOMSERVICE resource definition that was used to process this task.

383 (TYPE-C, 'WBSVCENM', 32 BYTES)
For web service applications, the name of the WEBSERVICE resource definition that was used to process this task.

384 (TYPE-C, 'WBSVOPNM', 64 BYTES)
For web service applications, the first 64 bytes of the web service operation name.

385 (TYPE-C, 'WBPROGNM', 8 BYTES)
For CICS web support, the name of the program from the URIMAP resource definition that was used to provide the application-generated response to the HTTP request processed by this task.

386 (TYPE-A, 'WBSFCRCT', 4 BYTES)
The number of **EXEC CICS SOAPFAULT CREATE** commands issued by the user task.

387 (TYPE-A, 'WBSFTOCT', 4 BYTES)
The total number of **EXEC CICS SOAPFAULT ADD**, **CREATE**, and **DELETE** commands issued by the user task.

388 (TYPE-A, 'WBISSFCT', 4 BYTES)
The total number of SOAP faults received in response to the **EXEC CICS INVOKE SERVICE** and **EXEC CICS INVOKE WEBSERVICE** commands issued by the user task.

390 (TYPE-A, 'WBSREQBL', 4 BYTES)
For web service applications, the SOAP request body length.

392 (TYPE-A, 'WBSRSPBL', 4 BYTES)
For web service applications, the SOAP response body length.

412 (TYPE-A, 'MLXSSTDL', 4 BYTES)
The total length of the documents that were parsed using the z/OS XML System Services parser.

413 (TYPE-A, 'MLXMLTCT', 4 BYTES)
The number of **EXEC CICS TRANSFORM** commands issued by the user task.

420 (TYPE-A, 'WSACBLCT', 4 BYTES)
The number of **EXEC CICS WSACONTEXT BUILD** commands issued by the user task.

421 (TYPE-A, 'WSACGTCT', 4 BYTES)
The number of **EXEC CICS WSACONTEXT GET** commands issued by the user task.

422 (TYPE-A, 'WSAEPCCT', 4 BYTES)
The number of **EXEC CICS WSAEPR CREATE** commands issued by the user task.

423 (TYPE-A, 'WSATOTCT', 4 BYTES)
The total number of EXEC CICS WS-Addressing commands issued by the user task.

424 (TYPE-A, 'WBJSNRQL', 4 BYTES)
For JSON web service applications, the JSON message request length.

425 (TYPE-A, 'WBJSNRPL', 4 BYTES)
For JSON web service applications, the JSON message response length.

Monitoring fields for URIMAP usage types

Table 33 shows which fields in the DFHWEBB group apply to each kind of service provided by URIMAP resource definitions, as determined by the USAGE attribute and other attributes of the URIMAP resource definition.

Table 33. Monitoring fields for URIMAP usage types

Field id	USAGE (PIPELINE): web service	USAGE (ATOM): Atom feed	USAGE (SERVER): CICS web support dynamic response (with program)	USAGE (SERVER): CICS web support static response (with zFS file or document template)
380 WBURIMNM	URIMAP resource definition name	URIMAP resource definition name	URIMAP resource definition name	URIMAP resource definition name
381 WBPIPLNM	PIPELINE resource definition name	null	null	null
382 WBATMSNM	null	ATOMSERVICE resource definition name	null	null
383 WBSVCENM	WEBSERVICE resource definition name	null	null	null
384 WBSVOPNM	WEBSERVICE operation name	null	null	null
385 WBPROGNM	null	null	PROGRAM resource definition name	null

Exception class data: listing of data fields

The exception class data is listed in the order in which it appears in the exception data section of a monitoring record.

Exception records are fixed format. The format of the exception data section of a monitoring record can be mapped by the DSECT MNEXCDS.

EXCMNTRN (TYPE-C, 4 BYTES)
Transaction identification.

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 identifier 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 low-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)
z/OS Communications Server 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 34 on page 396.

EXCMNRID (TYPE-C, 8 BYTES)
Exception resource identification. The possible values for EXCMNRID are shown in Table 34 on page 396.

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

X'0001'
Exception because of a wait (EXCMNWT)

X'0002'
Exception because of a buffer wait (EXCMNBWT)

X'0003'
Exception because of a string wait (EXCMNSWT)

X'0004'
Exception because of a policy threshold has been exceeded (EXCMNPOL)

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

EXCMNSRV (TYPE-C, 8 BYTES)
z/OS Workload Manager Service Class name for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active z/OS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

EXCMNRPT (TYPE-C, 8 BYTES)
z/OS Workload Manager Report Class name for this transaction. This field is null if there are no transaction classification rules defined for CICS subsystems in the active z/OS Workload Manager (WLM) service policy, or if the transaction was WLM-classified in another CICS region.

EXCMNNPX (TYPE-C, 20 BYTES)
Fully qualified name by which the originating system is known to the z/OS Communications Server 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 passing bytes (X'00') are present at the right end of the name.

If the originating terminal is a z/OS Communications Server device across an ISC APPC or IRC link, the NETNAME is the *networkid.LUname*. If the terminal is not a z/OS Communications Server device, 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-z/OS Communications Server terminal originators.

When the originator is communicating over an external CICS interface (EXCI) session, the name is a concatenation of the following information that is derived from the originating system:

'DFHEXCIU 8 bytes	. 1 byte	MVS Id 4 bytes	Address space Id (ASID)' 4 bytes

That is, the name is a 17-byte LU name that consists of the following:
- An 8-byte eye-catcher set to 'DFHEXCIU'.
- A 1-byte field containing a period (.).
- A 4-byte field containing the MVS ID, 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.

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

The first six bytes of this field are a binary value derived from the clock of the originating system and wrapping round at intervals of several months. The last two bytes of this field are for the period count. These bytes can change during the life of the task as a result of syncpoint activity.

Note: When MRO or ISC is used, the EXCMNNSX field must be combined with the EXCMNNPX field to uniquely identify a task, because the EXCMNNSX field is unique only to the originating CICS system.

EXCMNTRF (TYPE-C, 8 BYTES)
Transaction flags; a string of 64 bits used for signaling transaction definition and status information. For details, see field 164 (TRANFLAG) in performance data group DFHTASK.

EXCMNFCN (TYPE-C, 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. The transaction facility type (if any) can be identified by using byte 0 of the transaction flags field, EXCMNTRF.

EXCMNCPN (TYPE-C, 8 BYTES)
The name of the currently running program for this user task when the exception condition occurred.

EXCMNBTR (TYPE-C, 4 BYTES)
3270 Bridge transaction identification.

EXCMNURI (TYPE-C, 16 BYTES)
RRMS/MVS unit-of-recovery ID (URID).

EXCMNRIL (TYPE-A, 4 BYTES)
Exception resource ID length.

EXCMNRIX (TYPE-C, 256 BYTES)
Exception resource ID (extended).

EXCMNNID (TYPE-C, 8 BYTES)
NETID if a network qualified name has been received from z/OS Communications Server. For a z/OS Communications Server resource when the network qualified name has not yet been received, NETID is eight blanks. In all other cases, this field is nulls.

EXCMNRLU (TYPE-C, 8 BYTES)
Real network name if a network qualified name has been received from z/OS Communications Server. In all other cases, this field is the same as LUNAME (field id 111). For non-z/OS Communications Server resources, this field is nulls.

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

Table 34. 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	MEANING
EXCMNPOL	'DATABASE'	rule_id **1**	A database request policy rule has been exceeded
EXCMNPOL	'FILE'	rule_id **1**	A file request policy rule has been exceeded
EXCMNPOL	'PROGRAM'	rule_id **1**	A program request policy rule has been exceeded
EXCMNPOL	'START'	rule_id **1**	A start request policy rule has been exceeded
EXCMNPOL	'STORAGE'	rule_id **1**	A storage or storage request policy rule has been exceeded
EXCMNPOL	'SYNCPT'	rule_id **1**	A syncpoint request policy rule has been exceeded
EXCMNPOL	'TDQUEUE'	rule_id **1**	A TD Queue request policy rule has been exceeded
EXCMNPOL	'TIME'	rule_id **1**	A time policy rule has been exceeded
EXCMNPOL	'TSQUEUE'	rule_id **1**	A TS Queue bytes or TS Queue request policy rule has been exceeded

Table 34. Possible values of EXCMNTYP, EXCMNRTY, and EXCMNRID (continued). The relationship between exception type, resource type, and resource identification.

EXCMNTYP Exception type	EXCMNRTY Resource type	EXCMNRID Resource ID	MEANING
EXCMNWT	'CFDTLRSW'	poolname	Wait for coupling facility data tables locking (request) slot
EXCMNWT	'CFDTPOOL'	poolname	Wait for coupling facility data tables non-locking (request) slot
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	'STORAGE'	'GCDSA'	Wait for GCDSA storage
EXCMNWT	'STORAGE'	'GUDSA'	Wait for GUDSA storage
EXCMNWT	'STORAGE'	'GSDSA'	Wait for GSDSA 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

Note:

1 **'rule_id'**
The name of a rule_id is a concatenation of the bundle_id, policy name, and rule name: **<bundle id>.<policy name>.<rule name>**.

EXCMNRID contains the first 8 characters of the rule_id, and the full name is in EXCMNRIX.

Related concepts:

"Exception class data" on page 298
Exception class monitoring data is information on CICS resource shortages that are suffered by a transaction, such as queuing for file strings, or waiting for temporary storage. This data highlights possible problems in CICS system operation, and is intended to help you identify system constraints that affect the performance of your transactions. CICS writes one exception record for each exception condition that occurs.

Transaction resource class data: Listing of data fields

The transaction resource class data is listed in the order in which it appears in the transaction resource data section of a monitoring record.

All the transaction resource data records produced by a single CICS run have the same format, with a resource record header followed by a resource data section for each resource being monitored. The format of the transaction resource data section of a monitoring record can be mapped by the DSECT DFHMNRDS.

Header fields

These fields are the transaction header fields in a transaction resource monitoring record.

MNR_ID_TRANID (TYPE-C, 4 BYTES)
Transaction identifier.

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

MNR_ID_USERID (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.

MNR_ID_STYPE (TYPE-C, 4 BYTES)
Transaction start type. The high-order byte (0 and 1) can have one of the following values:

"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 the transient data trigger level

"U " Attached by a user request

"TP" Attached from a terminal TCTTE transaction ID

"SZ" Attached by the Front End Programming Interface (FEPI).

MNR_ID_START (TYPE-T, 8 BYTES)
Start time of the transaction.

MNR_ID_STOP (TYPE-T, 8 BYTES)
Stop time of the transaction.

MNR_ID_TASKNO (TYPE-A, 4 BYTES)
The transaction identification number (the task number allocated to the transaction at task attach).

MNR_ID_LUNAME (TYPE-C, 8 BYTES)
z/OS Communications Server logical unit name (if available) of the terminal associated with this transaction. If the task is running in an application-owning or file-owning region, the LUNAME is the generic applid of the originating connection for MRO, LUTYPE6.1, and LUTYPE6.2 (APPC). The LUNAME is blank if the originating connection is an external CICS interface (EXCI).

MNR_ID_PGMNAME (TYPE-C, 8 BYTES)
The name of the first program invoked at attach-time. For more information, see performance data field 071 (PGMNAME) in the DFHPROG group.

MNR_ID_UOW_PX (TYPE-C, 20 BYTES)
This field contains the same information as the performance data field NETUOWPX. For the details, see performance data field 097 (NETUOWPX) in the DFHTASK group.

MNR_ID_UOW_SX (TYPE-C, 8 BYTES)
This field contains the same information as the performance class data field NETUOWSX. For the details, see performance data field 098 (NETUOWSX) in the DFHTASK group.

MNR_ID_RSYSID (TYPE-C, 4 BYTES)
The name (system ID) of the remote system to which this transaction was routed, either statically or dynamically. For more information, see performance data field 130 (RSYSID) in the DFHCICS group.

MNR_ID_TRN_FLAGS (TYPE-A, 8 BYTES)
Transaction flags, a string of 64 bits used for signaling transaction definition and status information. For the details, see performance data field 164 (TRANFLAG) in the DFHTASK group.

MNR_ID_FCTYNAME (TYPE-C, 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. You can identify the transaction facility type (if any) using byte 0 of the transaction flags (MNR_ID_TRN_FLAGS) field. For details, see performance data field 163 (FCTYNAME) in the DFHTASK group.

MNR_ID_RTYPE (TYPE-C, 4 BYTES)
Transaction resource monitoring record type (low-order byte-3). Currently this record type can have only one value, T, indicating a record output for task termination. For more information about record types, see performance data field 112 (RTYPE) in the DFHCICS group.

MNR_ID_TERMINFO (TYPE-A, 4 BYTES)
Terminal or session information for the task principal facility. For more information about terminal information, see performance data field 165 (TERMINFO) in the DFHTERM group.

MNR_ID_TERMCNNM (TYPE-C, 4 BYTES)
Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (system ID). For more information, see performance data field 169 (TERMCNNM) in the DFHTERM group.

MNR_ID_RES_FLAGS (TYPE-A, 4 BYTES)
Resource flags, a string of 32 bits used for signaling resource status information.

Byte 0 Resource status information:

Bit 0 The maximum number of files to be monitored (defined in the MCT) has been exceeded by the transaction (X'80')

Bit 1 The maximum number of temporary storage queues to be monitored (defined in the MCT) has been exceeded by the transaction (X'40')

Bit 2 The maximum number of distributed program link requests to be monitored (defined in the MCT) has been exceeded by the transaction (X'20')

Bits 3-7
Reserved.

Bytes 1-3
Reserved.

MNR_ID_ISIPICNM (TYPE-C, 8 BYTES)
The name of the IPIC (IPCONN) entry of the TCP/IP service that attached the user task. For more information, see field 305 (ISIPICNM) in the DFHSOCK performance-class data group.

MNR_ID_CLIPADDR (TYPE-C, 40 BYTES)
The IP address of the originating client or Telnet client. For more information, see field 318 (CLIPADDR) in the DFHSOCK performance-class data group.

MNR_ID_ORIGIN_NETWKID (TYPE-C, 8 BYTES)
The network identifier from which this work request (transaction) originated. For more information, see field 359 (ONETWKID) in the DFHCICS performance data group.

MNR_ID_ORIGIN_APPLID (TYPE-C, 8 BYTES)
The applid of the CICS region where this work request (transaction) originated; for example, the region in which the CWXN task ran. For more information, see field 360 (OAPPLID) in the DFHCICS performance data group.

MNR_ID_ORIGIN_ATT_TIME (TYPE-T, 8 BYTES)
The time when the originating task, for example, the CWXN task, was started. For more information, see field 361 (OSTART) in the DFHCICS performance data group.

MNR_ID_ORIGIN_TRANNUM (TYPE-P, 4 BYTES)
The number of the originating task; for example, the CWXN task. For more information, see field 362 (OTRANNUM) in the DFHCICS performance data group.

MNR_ID_ORIGIN_TRANID (TYPE-C, 4 BYTES)
The transaction ID (TRANSID) of the originating task; for example, the CWXN task. For more information, see field 363 (OTRAN) in the DFHCICS performance data group.

MNR_ID_ORIGIN_USERID (TYPE-C, 8 BYTES)
The originating Userid-2 or Userid-1, for example, from CWBA, depending on the originating task. For more information, see field 364 (OUSERID) in the DFHCICS performance data group.

MNR_ID_ORIGIN_USER_CORR (TYPE-C, 64 BYTES)
The originating user correlator. For more information, see field 365 (OUSERCOR) in the DFHCICS performance data group.

MNR_ID_ORIGIN_TCPIPSERV (TYPE-C, 8 BYTES)
The name of the originating TCPIPSERVICE. For more information, see field 366 (OTCPSVCE) in the DFHCICS performance data group.

MNR_ID_ORIGIN_PORTNUM (TYPE-A, 4 BYTES)
The port number used by the originating TCPIPSERVICE. For more information, see field 367 (OPORTNUM) in the DFHCICS performance data group.

MNR_ID_ORIGIN_CLIPADDR (TYPE-C, 40 BYTES)
The IP address of the originating client or Telnet client. For more information, see field 372 (OCLIPADR) in the DFHCICS performance data group.

MNR_ID_ORIGIN_CLIPPORT (TYPE-A, 4 BYTES)
The TCP/IP port number of the originating client or Telnet client. For more information, see field 369 (OCLIPORT) in the DFHCICS performance data group.

MNR_ID_ORIGIN_TRANFLAG (TYPE-A, 8 BYTES)
The originating transaction flags. This 64-bit string is used for signaling transaction definition and status information. For more information, see field 370 (OTRANFLG) in the DFHCICS performance data group.

MNR_ID_ORIGIN_FCTYNAME (TYPE-C, 8 BYTES)
The facility name of the originating transaction. If the originating transaction is not associated with a facility, this field is null. For more information, see field 371 (OFCTYNME) in the DFHCICS performance data group.

MNR_PHD_NTWKID (TYPE-C, 8 BYTES)
The network identifier of the CICS system of an immediately previous task in another CICS region with which this task is associated. For more information, see field 373 (PHNTWKID) in the DFHCICS performance data group.

MNR_PHD_APPLID (TYPE-C, 8 BYTES)
The APPLID from previous hop data. This is the APPLID of the CICS system of a previous task in another CICS system with which this task is associated. For more information, see field 374 (PHAPPLID) in the DFHCICS performance data group. For more information about previous hop data, see Previous hop data characteristics in Getting started.

MNR_PHD_ATTACH_TIME (TYPE-T, 8 BYTES)
The start time of the immediately previous task in another CICS system with which this task is associated. For more information, see field 375 (PHSTART) in the DFHCICS performance data group.

MNR_PHD_TRANNUM (TYPE-P, 4 BYTES)
The task number of the immediately previous task in another CICS system with which this task is associated. For more information, see field 376 (PHTRANNO) in the DFHCICS performance data group.

MNR_PHD_TRANID (TYPE-C, 4 BYTES)
The transaction ID (TRANSID) of the immediately previous task in another CICS system with which this task is associated. For more information, see field 377 (PHTRAN) in the DFHCICS performance data group.

MNR_PHD_COUNT (TYPE-A, 4 BYTES)
The number of times there has been a request from one CICS system to another CICS region to initiate a task with which this task is associated. For more information, see field 378 (PHCOUNT) in the DFHCICS performance data group.

MNR_ID_TRNGRPID (TYPE-C, 28 BYTES)
The transaction group ID of the originating task.

File entry fields

These fields are in each file entry in a transaction resource monitoring record.

For information about transaction file accesses in performance class monitoring data, see DFHFILE group.

MNR_FILE_NAME (TYPE-C, 8 BYTES)
The CICS 8-character name of the file to which the following data fields refer.

MNR_FILE_GET (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of GET requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of GET requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_PUT (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of PUT requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of PUT requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_BRWSE (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of BROWSE requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of BROWSE requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_ADD (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of ADD requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of ADD requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_DEL (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of DELETE requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of DELETE requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_TOTAL (TYPE-S, 8 BYTES)
The total elapsed time that the user task waited for completion of all requests issued by the user task for this file. The count part of this field (the low-order 24 bits) contains the number of all requests issued against the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_FILE_AM_RQ (TYPE-A, 4 BYTES)
Number of times the user task called file access-method interfaces. See also performance data field 070 (FCAMCT) in the DFHFILE group.

MNR_FILE_IO_WT (TYPE-S, 8 BYTES)
The total I/O wait time on this file. The count part of this field (the low-order 24 bits) contains the number of requests issued against the file which waited for I/O.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_RLS_FILE_IO_WT (TYPE-S, 8 BYTES)
The elapsed time in which the user task waited for RLS file I/O on this file. The count part of this field (the low-order 24 bits) contains the number of requests issued against the RLS file which waited for I/O.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_CFDT_IO_WT (TYPE-S, 8 BYTES)
The elapsed time in which the user task waited for a data table access request to the coupling facility data table server to complete for this file. The count part of this field (the low-order 24 bits) contains the number of requests to the coupling facility data table server for the file.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

Temporary storage queue entry fields

These fields are in each temporary storage queue entry in a transaction resource monitoring record.

For information about transaction temporary storage queue accesses in performance class monitoring data, see DFHTEMP group.

MNR_TSQUEUE_NAME (TYPE-C, 16 BYTES)
The CICS 16-character name of the temporary storage queue to which the following data fields refer.

MNR_TSQUEUE_GET (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of GET requests issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of GET requests issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_PUT_AUX (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of PUT requests to auxiliary temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of PUT requests to auxiliary temporary storage issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_PUT_MAIN (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of PUT requests to

main temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of PUT requests to main temporary storage issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_TOTAL (TYPE-S, 8 BYTES)
The total elapsed time that the user task waited for completion of all requests issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of all requests issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_GET_ITEML (TYPE-A, 4 BYTES)
The total length of all items obtained from this temporary storage queue.

MNR_TSQUEUE_PUT_AUX_ITEML (TYPE-A, 4 BYTES)
The total length of all items written to the auxiliary temporary storage queue.

MNR_TSQUEUE_PUT_MAIN_ITEML (TYPE-A, 4 BYTES)
The total length of all items written to the main temporary storage queue.

MNR_TSQUEUE_IO_WT (TYPE-S, 8 BYTES)
The total I/O wait time on this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of requests issued against the temporary storage queue which waited for I/O.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_SHR_TSQUEUE_IO_WT (TYPE-S, 8 BYTES)
The total I/O wait time on the shared temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of requests issued against the shared temporary storage queue which waited for I/O.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_GET_SHR (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of GET requests to shared temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of GET requests to shared temporary storage issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_PUT_SHR (TYPE-S, 8 BYTES)
The elapsed time that the user task waited for completion of PUT requests to shared temporary storage, issued by the user task for this temporary storage queue. The count part of this field (the low-order 24 bits) contains the number of PUT requests to shared temporary storage issued against the temporary storage queue.

For more information, see Clocks and time stamps in Reference -> Monitoring and Transaction wait (suspend) times in Reference -> Monitoring.

MNR_TSQUEUE_GET_SHR_ITEML (TYPE-A, 4 BYTES)
The total length of all items obtained from this shared temporary storage queue.

MNR_TSQUEUE_PUT_SHR_ITEML (TYPE-A, 4 BYTES)
The total length of all items written to this shared temporary storage queue.

DPL entry fields

These fields are in each distributed program link entry in a transaction resource monitoring record.

For information about transaction program accesses in performance class monitoring data, see DFHPROG group.

MNR_DPL_PROGRAM_NAME (TYPE-C, 8 BYTES)
The name of the program to which the following data fields refer.

MNR_DPL_SYSID (TYPE-C, 4 BYTES)
The name of the remote system to which this program was routed for the distributed program link.

MNR_DPL_LINK_REQS (TYPE-C, 4 BYTES)
The number of distributed program link requests issued by the user task for this program and sysid combination.

Related concepts:

"Transaction resource class data" on page 299
Transaction resource class data provides additional transaction-level information about individual resources accessed by a transaction. Currently, the transaction resource class covers distributed program link, file, and temporary storage queue resources.

Identity class data: Listing of data fields

The identity class data is listed in the order in which it appears in the identity class data section of a monitoring record.

All the identity class data records produced by a single CICS run have the same format, with an identity record header followed by an identity data section for each transaction being monitored. The format of the identity class data section of a monitoring record can be mapped by the DSECT DFHMNIDS.

Header fields

These fields are the header fields in an identity class monitoring record.

MNI_ID_TRANID (TYPE-C, 4 BYTES)
Transaction identifier.

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

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

MNI_ID_STYPE (TYPE-C, 4 BYTES)
Transaction start type. The high-order bytes (0 and 1) can have one of the following values:

"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 the transient data trigger level

"U " Attached by a user request

"TP" Attached from a terminal TCTTE transaction ID

"SZ" Attached by the Front End Programming Interface (FEPI)

MNI_ID_START (TYPE-T, 8 BYTES)
Start time of the transaction.

MNI_ID_STOP (TYPE-T, 8 BYTES)
Stop time of the transaction.

MNI_ID_TASKNO (TYPE-A, 4 BYTES)
The transaction identification number (the task number allocated to the transaction at task attach).

MNI_ID_LUNAME (TYPE-C, 8 BYTES)
z/OS Communications Server logical unit name (if available) of the terminal associated with this transaction. If the task is running in an application-owning or file-owning region, the LUNAME is the generic applid of the originating connection for MRO, LUTYPE6.1, and LUTYPE6.2 (APPC). The LUNAME is blank if the originating connection is an external CICS interface (EXCI).

MNI_ID_PGMNAME (TYPE-C, 8 BYTES)
The name of the first program called at attach-time. For more information, see field 071 (PGMNAME) in the DFHPROG performance data group.

MNI_ID_UOW_PX (TYPE-C, 20 BYTES)
This field contains the same information as the performance class data field NETUOWPX. See NETUOWPX, in group DFHTASK for details.

MNI_ID_UOW_SX (TYPE-C, 8 BYTES)
This field contains the same information as the performance class data field NETUOWSX. See NETUOWSX, in group DFHTASK for details.

MNI_ID_RSYSID (TYPE-C, 4 BYTES)
The name (system ID) of the remote system to which this transaction was routed, either statically or dynamically. For more information, see field 130 (RSYSID) in the DFHCICS performance data group.

MNI_ID_TRN_FLAGS (TYPE-A, 8 BYTES)
Transaction flags, a string of 64 bits used for signaling transaction definition and status information. For details, see field 164 (TRANFLAG) in the DFHTASK performance data group.

MNI_ID_FCTYNAME (TYPE-C, 4 BYTES)
Transaction facility name. This field is null if the transaction is not associated with a facility. You can identify the transaction facility type (if any) using byte 0 of the transaction flags (MNR_ID_TRN_FLAGS) field. For details, see field 163 (FCTYNAME) in the DFHTASK performance data group.

MNI_ID_RTYPE (TYPE-C, 4 BYTES)
Transaction resource monitoring record type (low-order byte 3). Currently this

record type can have only one value, T, indicating a record produced for task termination. For more information about record types, see field 112 (RTYPE) the DFHCICS performance data group.

MNI_ID_TERMINFO (TYPE-A, 4 BYTES)
Terminal or session information for the task principal facility. For more information about terminal information, see field 165 (TERMINFO) in the DFHTERM performance data group.

MNI_ID_TERMCNNM (TYPE-C, 4 BYTES)
Terminal session connection name. If the terminal facility associated with this transaction is a session, this field is the name of the owning connection (system ID). For more information, see field 169 (TERMCNNM) in the DFHTERM performance data group.

MNI_ID_ISIPICNM (TYPE-C, 8 BYTES)
The name of the IPIC (IPCONN) entry of the TCP/IP service that attached the user task. For more information, see field 305 (ISIPICNM) in the DFHSOCK performance-class data group.

MNI_ID_CLIPADDR (TYPE-C, 40 BYTES)
The IP address of the originating client or Telnet client. For more information, see field 318 (CLIPADDR) in the DFHSOCK performance-class data group.

MNI_ID_ORIGIN_NETWKID (TYPE-C, 8 BYTES)
The network identifier from which this work request (transaction) originated. For more information, see field 359 (ONETWKID) in the DFHCICS performance data group.

MNI_ID_ORIGIN_APPLID (TYPE-C, 8 BYTES)
The applid of the CICS region where this work request (transaction) originated; for example, the region in which the CWXN task ran. For more information, see field 360 (OAPPLID) in the DFHCICS performance data group.

MNI_ID_ORIGIN_ATT_TIME (TYPE-T, 8 BYTES)
The time when the originating task, for example, the CWXN task, was started. For more information, see field 361 (OSTART) in the DFHCICS performance data group.

MNI_ID_ORIGIN_TRANNUM (TYPE-P, 4 BYTES)
The number of the originating task; for example, the CWXN task. For more information, see field 362 (OTRANNUM) in the DFHCICS performance data group.

MNI_ID_ORIGIN_TRANID (TYPE-C, 4 BYTES)
The transaction ID (TRANSID) of the originating task; for example, the CWXN task. For more information, see field 363 (OTRAN) in the DFHCICS performance data group.

MNI_ID_ORIGIN_USERID (TYPE-C, 8 BYTES)
The originating Userid-2 or Userid-1, for example, from CWBA, depending on the originating task. For more information, see field 364 (OUSERID) in the DFHCICS performance data group.

MNI_ID_ORIGIN_USER_CORR (TYPE-C, 64 BYTES)
The originating user correlator. For more information, see field 365 (OUSERCOR) in the DFHCICS performance data group.

MNI_ID_ORIGIN_TCPIPSERV (TYPE-C, 8 BYTES)
The name of the originating TCPIPSERVICE. For more information, see field 366 (OTCPSVCE) in the DFHCICS performance data group.

MNI_ID_ORIGIN_PORTNUM (TYPE-A, 4 BYTES)
The port number used by the originating TCPIPSERVICE. For more information, see field 367 (OPORTNUM) in the DFHCICS performance data group.

MNI_ID_ORIGIN_CLIPADDR (TYPE-C, 40 BYTES)
The IP address of the originating client or Telnet client. For more information, see field 372 (OCLIPADR) in the DFHCICS performance data group.

MNI_ID_ORIGIN_CLIPPORT (TYPE-A, 4 BYTES)
The TCP/IP port number of the originating client or Telnet client. For more information, see field 369 (OCLIPORT) in the DFHCICS performance data group.

MNI_ID_ORIGIN_TRANFLAG (TYPE-A, 8 BYTES)
The originating transaction flags. This 64-bit string is used for signaling transaction definition and status information. For more information, see field 370 (OTRANFLG) in the DFHCICS performance data group.

MNI_ID_ORIGIN_FCTYNAME (TYPE-C, 8 BYTES)
The facility name of the originating transaction. If the originating transaction is not associated with a facility, this field is null. For more information, see field 371 (OFCTYNME) in the DFHCICS performance data group.

MNI_ID_PHD_NTWKID (TYPE-C, 8 BYTES)
The network identifier of the CICS system of an immediately previous task in another CICS system with which this task is associated. For more information, see field 373 (PHNTWKID) in the DFHCICS performance data group.

MNI_ID_PHD_APPLID (TYPE-C, 8 BYTES)
The APPLID from previous hop data. This is the APPLID of the CICS system of a previous task in another CICS system with which this task is associated. For more information, see field 374 (PHAPPLID) in the DFHCICS performance data group. For more information about previous hop data, see Previous hop data characteristics in Getting started.

MNI_ID_PHD_START_TIME (TYPE-T, 8 BYTES)
The start time of the immediately previous task in another CICS system with which this task is associated. For more information, see field 375 (PHSTART) in the DFHCICS performance data group.

MNI_ID_PHD_TRANNO (TYPE-P, 4 BYTES)
The task number of the immediately previous task in another CICS system with which this task is associated. For more information, see field 376 (PHTRANNO) in the DFHCICS performance data group.

MNI_ID_PHD_TRANID (TYPE-C, 4 BYTES
The transaction ID (TRANSID) of the immediately previous task in another CICS system with which this task is associated. For more information, see field 377 (PHTRAN) in the DFHCICS performance data group.

MNI_ID_PHD_COUNT (TYPE-A, 4 BYTES)
The number of times there has been a request from one CICS system to another CICS system to initiate a task with which this task is associated. For more information, see field 378 (PHCOUNT) in the DFHCICS performance data group.

Data entry fields

Each identity record consists of an identity record header, an identity record identification section, and two identity data entries (an entry for the distinguished name and an entry for the realm). Each identity data entry consists of an entry identifier field, an entry length field, and a variable length entry field.

MNI_ENTRY_IDENT
Data entry identifier.

MNI_ENTRY_LENGTH
Length of the data entry that is specified by the data entry identifier.

MNI_ENTRY_FIELD
Data entry field.

Table 35. Identity record data entry fields

Data entry identifier decimal (hexadecimal)	Data entry length	Format	Description
1 (1)	1 - 246	UTF-8	A distinguished name, which uniquely identifies the user.
2 (2)	1 - 255	UTF-8	A realm, which identifies the set of resources to which the authentication information requested (that is, the user ID and password) applies.

Related concepts:

"Identity class data" on page 301
Identity class data provides enhanced audit information by capturing identity propagation data (an X.500 distinguished name and associated realm) from a client system across a network for eligible transactions.

Part 4. CICS statistics

As events occur, CICS produces information that is available to you as system and resource statistics. Statistics are collected during CICS online processing for later offline analysis. These topics provide information about the statistics produced by CICS and the ways to report them.

Chapter 31. Introduction to CICS statistics

CICS produces five types of statistics: interval statistics, end-of-day statistics, requested statistics, requested reset statistics, and unsolicited statistics.

The CICS statistics domain writes statistics records to a System Management Facilities (SMF) data set. The records are of SMF type 110, subtype 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.

The global and specific, or resource, statistics for each resource type are each mapped by a different DSECT. Programming information about the formats of CICS statistics records is given in Writing statistics collection and analysis programs in the *CICS Customization Guide*.

Interval statistics

Interval statistics are gathered by CICS during a specified interval. You can change the interval value by using the **STATINT** system initialization parameter. CICS writes the interval statistics to the SMF data set automatically at the expiry of the interval if any of the following conditions are satisfied:
- Statistics recording status was set to ON by the **STATRCD** system initialization parameter (and has not then been set to OFF by a **EXEC CICS SET STATISTICS RECORDING** command). The default is STATRCD=OFF.
- The RECORDING option of the **EXEC CICS SET STATISTICS** command is set to ON.

End-of-day statistics

End-of-day statistics are a special case of interval statistics where all statistics counters are collected and reset. There are three ways to get end-of-day statistics:
- 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. The default end-of-day value is 000000 (midnight). Change the end-of-day value by using the **STATEOD** system initialization parameter, or using 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 items:
- The system initialization parameter, **STATRCD**
- The RECORDING option of **EXEC CICS SET STATISTICS**.

The statistics that are written to the SMF data set are the statistics collected since the last event that involved a reset. The following are examples of resets:
- At CICS startup
- Issue of **RESETNOW RECORDNOW** or **EXEC CICS STATISTICS** commands

- Interval statistics

Requested statistics

Requested statistics are produced by using one of the following commands:
- EXEC CICS PERFORM STATISTICS RECORD
- EXEC CICS SET STATISTICS RECORDNOW
- CEMT PERFORM STATISTICS RECORD

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.

Requested reset statistics

Requested reset statistics differ from requested statistics in that all statistics are collected and statistics counters are reset. You can reset the statistics counters using the following commands:
- EXEC CICS PERFORM STATISTICS RECORD ALL RESETNOW
- EXEC CICS SET STATISTICS ON|OFF RESETNOW RECORDNOW
- CEMT PERFORM STATISTICS RECORD ALL RESETNOW

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

You can also invoke requested reset statistics when changing the recording status from ON to OFF, or from OFF to ON, using **EXEC CICS SET STATISTICS ON|OFF RECORDNOW RESETNOW**. It is valid to specify the RECORDNOW RESETNOW options only when there is a genuine change of status from STATISTICS ON to OFF, or from OFF to ON. In other words, coding **EXEC CICS SET STATISTICS ON RECORDNOW RESETNOW** when statistics recording is already ON causes an error response.

Using the **SET STATISTICS RESETNOW** 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 to ON. To set the statistics recording status to ON or OFF, use either the RECORDING option on this command or the **STATRCD** system initialization parameter. Statistics are always written, and counts reset, at the end of day.

The following figure summarizes the statistics reset functions.

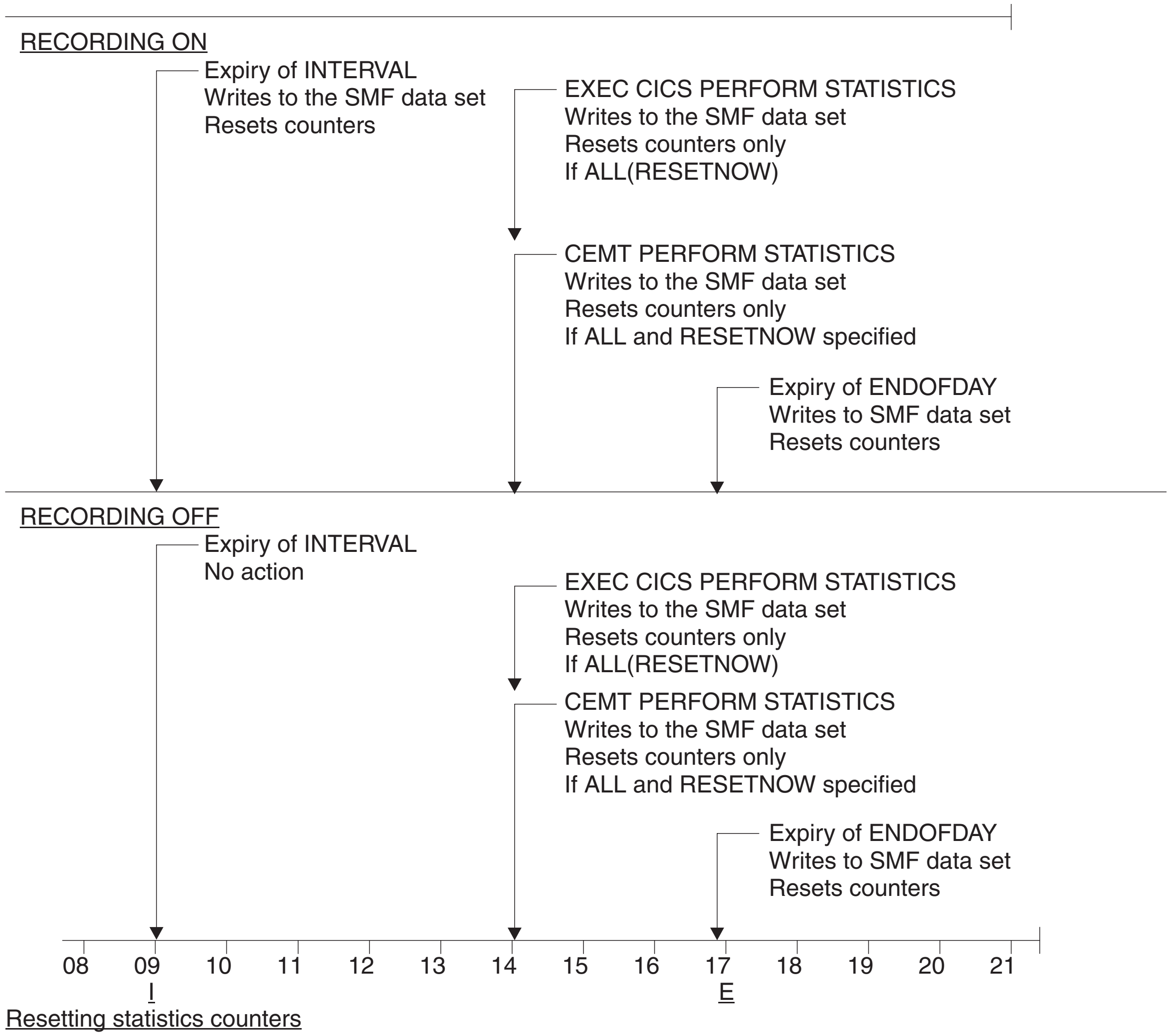


Figure 57. Summary of statistics reset functions

Unsolicited statistics

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.

To ensure that accurate statistics are recorded, unsolicited statistics (USS) must be collected. An unsolicited record resets the statistics fields it contains. In particular, during a normal CICS shutdown, files are closed before the end-of-day statistics are gathered. Closing files before the end-of-day statistics are gathered means that file and LSRpool end-of-day statistics are zero, while the correct values are recorded as unsolicited statistics.

Unsolicited statistics are produced for the following resources:

Atom feeds
: Whenever an ATOMSERVICE resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

Autoinstalled terminals
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 **AILDELAY** system initialization parameter.

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

CAPTURESPEC resources
Whenever a CAPTURESPEC resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

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

DB2 Whenever CICS disconnects from DB2, CICS collects the statistics for the DB2 connection and all DB2ENTRY resources covering the period from the last interval.

Whenever a DB2ENTRY is discarded, CICS collects the statistics for that DB2ENTRY covering the period from the last interval.

DOCTEMPLATE resources
Whenever a document template is discarded, CICS collects the statistics for that template covering the period from the last interval.

EPADAPTER resources
When an EPADAPTER resource is disabled, CICS collects the statistics for that resource covering the period from the last interval.

EVENTBINDING resources
Whenever an EVENTBINDING resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

FEPI connection
Unsolicited connection statistics are produced when a connection is destroyed.

FEPI pools
Unsolicited pool statistics are produced when a pool is discarded by using the DISCARD POOL or DELETE POOL command.

FEPI targets
Unsolicited target statistics are produced when a target is destroyed or removed from a pool.

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

IPCONN resources
Whenever an IPIC connection is discarded, CICS collects the statistics for that IPCONN resource covering the period from the last interval.

Journalnames
Unsolicited journalname statistics are produced when a JOURNALNAME resource is discarded.

JVMSERVER resources
When a JVMSERVER resource is disabled, CICS collects the statistics for that resource covering the period from the last interval.

LIBRARY resources
Whenever a LIBRARY resource is disabled, CICS collects the statistics for that definition covering the period from the last interval.

Logstreams
Unsolicited logstream statistics are produced when the logstream is discarded from the MVS system logger.

LSR pools
When CICS closes a file that is in an LSR pool, CICS collects the statistics for the LSR pool. The following peak values are reset 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 LSR pool is created (when the first file is opened) until the LSR pool is deleted (when the last file is closed).

MQCONN resources
Whenever a WebSphere MQ connection is disconnected, CICS collects the statistics for that MQCONN resource covering the period from the last interval.

PIPELINE resources
Whenever a PIPELINE resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

PROGRAM resources
When an installed program is discarded, CICS collects the statistics covering the installed period since the last interval.

Program definitions
When an installed PROGRAM resource definition is discarded, CICS collects the statistics covering the installed period since the last interval.

TCP/IP Services
Whenever CICS closes a TCP/IP service, CICS collects the statistics covering the period since the last interval.

Transactions
When an installed TRANSACTION resource definition is discarded, CICS collects the statistics covering the installed period since the last interval.

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

Transient data queues
Unsolicited transient data queue statistics are produced when a transient data queue is discarded or when an extrapartition transient data queue is closed.

URIMAP resources
Whenever a URIMAP resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

WEBSERVICE resources
Whenever a WEBSERVICE resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

XMLTRANSFORM resources
Whenever an XMLTRANSFORM resource definition is discarded, CICS collects the statistics for that resource covering the period from the last interval.

Statistics for private resources for applications deployed on platforms

For resource types that are supported as private resources for applications deployed on platforms, separate statistics records are written for the public resources and for the private resources, each mapped by a different DSECT.

When you use the **EXEC CICS PERFORM STATISTICS RECORD** command to write resource statistics, use the same resource type keyword whether the resource is public or private. If a resource is a public resource, the public DSECT is used to map its data, and if a resource is a private resource, the private DSECT is used to map its data.

When you use the **EXEC CICS EXTRACT STATISTICS** or **EXEC CICS COLLECT STATISTICS** command to request resource statistics for a specific resource of a resource type that is supported as a private resource, the command operates according to the context in which the task is running.

- If the command is issued from a public program, statistics are returned for the named public resource.
- If the command is issued from a program that is part of an application deployed on a platform, so is running with an application context, the private resources for the application are searched first for the named resource. If a private resource is not found, statistics are returned for the named public resource.
- For the **EXEC CICS EXTRACT STATISTICS** command only, you can specify a different application context to be searched for private resources. When you request statistics for a different application, if a private resource is not found for that application, no statistics are returned.

Programs that are declared as application entry points are identified by a field in the DSECTs for public and private program definitions (PROGRAMDEF statistics keyword) and JVM programs (JVMPROGRAM keyword). When interval statistics, end-of-day statistics, requested statistics, requested reset statistics, or unsolicited statistics are produced for a program definition or JVM program that is declared as an application entry point, two statistics records are written, one mapped by the DSECT for public resources, and one mapped by the DSECT for private resources. For the program statistics that are produced by the loader domain (PROGRAM keyword), application entry points are not identified, and only one private program statistics record is written.

When you use the **EXEC CICS EXTRACT STATISTICS** or **EXEC CICS COLLECT STATISTICS** command to return statistics for a specified program that is declared as an application entry point, only one statistics record is returned. If the command is issued in or for an application context, and the program was defined as a private resource for the application, the DSECT for private resources is used to format the data, even if the program has currently been promoted to a public program in order to make the application entry point available.

Reset characteristics of statistics counters

When statistics are written to the SMF data set, the counters might be reset.

- Reset to zero
- Reset to 1
- Reset to current values (this applies to peak values)
- Other alternative values.

For detailed information about the reset characteristics, see "CICS statistics in DSECTs and DFHSTUP report" on page 420.

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 58, 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.

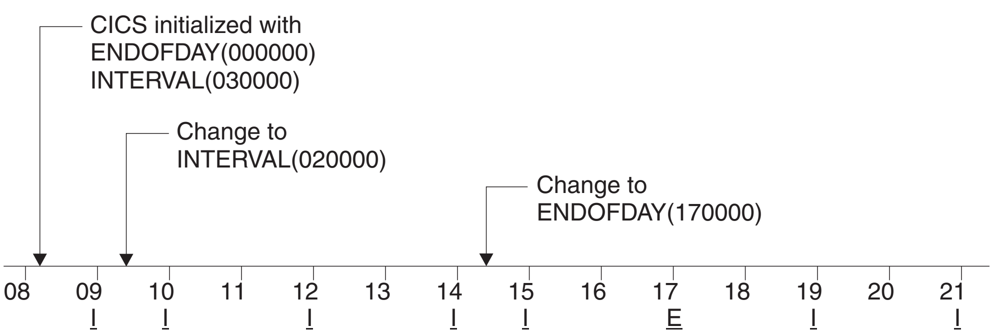


Figure 58. Resetting statistics counters

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$STED which changes the end-of-day, and thus each interval expiry boundary, by a few seconds. See Stagger end-of-day time sample utility program the *CICS Operations and Utilities Guide* for further information about DFH$STED.

Setting STATRCD=OFF reduces the number of times that statistics are written to the SMF data set and the counters are reset to the end-of-day, unsolicited, and requested reset only."

Processing CICS statistics

CICS has several utilities and sample programs to help you process and analyze statistics. You can also use other products to access the statistics data and analyze your CICS regions.

About this task

You can process CICS statistics in the following ways:

Procedure

- Use the CICS DFHSTUP offline utility. DFHSTUP prepares and prints reports offline, using the CICS statistics data recorded on the MVS system management facilities (SMF) SYS1.MANx data sets. For guidance about retrieving CICS statistics from SMF, and about running DFHSTUP, see Statistics utility program (DFHSTUP) in the *CICS Operations and Utilities Guide*.
- Use the sample statistics program (DFH0STAT). You can use the statistics sample program, DFH0STAT, to produce online reports from the CICS statistics data. The program demonstrates the use of the **EXEC CICS INQUIRE**, **EXEC CICS COLLECT STATISTICS**, and **EXEC CICS EXTRACT STATISTICS** commands to produce an analysis of a CICS system. You can use the sample program as provided or modify it to suit your needs. For more information, see "The sample statistics program, DFH0STAT" on page 423.
- Use CICS Performance Analyzer to produce reports and extracts using CICS Monitoring Facility (CMF) and CICS statistics SMF 110 records. For more information, see "CICS Performance Analyzer for z/OS (CICS PA)" on page 27.
- Use Tivoli Decision Support to process CICS SMF records to produce joint reports with data from other SMF records. For more information, see "Performance measuring with Tivoli Decision Support for z/OS" on page 37.
- Create your own statistics reports using the DFHSTUP extract statistics reporting facility. This facility provides a DFHSTUP exit that sends CICS statistics data to a user program that can process statistics records to create tailored reports. For guidance about using the extract reporting facility, see The DFHSTUP extract statistics reporting function
- 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 CICS statistics record format in the *CICS Customization Guide*.

CICS statistics in DSECTs and DFHSTUP report

The main reference information for the CICS statistics lists them as they are presented in the report from the DFHSTUP utility, and in the statistics DSECTs.

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 DFHSTIDS copybook. Programming information about the formats of CICS statistics records is given in Writing statistics collection and analysis programs in the *CICS Customization Guide*.

Statistics areas are listed alphabetically. Each area of CICS statistics is listed in the following format:

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 mapping this data.	**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 alternative value (these are documented).

For resource types that are supported as private resources for applications deployed on platforms, statistics for both public and private resources are reported in the output from the DFHSTUP utility. The statistics for public resources and the statistics for private resources are mapped by separate DSECTs, and are also reported separately by DFHSTUP.

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 Statistics utility program (DFHSTUP) in the *CICS 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 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.

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 36. CICS statistics areas

Statistic type	Topic
Atom feeds	"Atom feed statistics" on page 429
Autoinstall global statistics	"Autoinstall statistics" on page 434
Bundle	"BUNDLE statistics" on page 439
Capture specification	"CAPTURESPEC statistics" on page 499
CICS DB2	"CICS DB2 statistics" on page 441
DBCTL session termination	"DBCTL session termination statistics" on page 465
Dispatcher domain	"Dispatcher domain statistics" on page 468
Document templates	"Document template statistics" on page 485
Dump domain — system dump	"Dump domain: System dump statistics" on page 490
Dump domain — transaction dump	"Dump domain: Transaction dump statistics" on page 493
Enqueue domain	"Enqueue domain statistics" on page 495
EP adapter	"EPADAPTER statistics" on page 501
Event binding	"EVENTBINDING statistics" on page 503
Event process	"EVENTPROCESS statistics" on page 507
Front end programming interface (FEPI)	"Front end programming interface (FEPI) statistics" on page 511
File control	"File control statistics" on page 517
IPCONN	"IPCONN statistics" on page 561
ISC/IRC system and mode entry	"ISC/IRC system and mode entry statistics" on page 535
ISC/IRC attach time entry	"ISC/IRC attach time entry statistics" on page 559
Journalname	"Journalname statistics" on page 573
JVM server	"JVMSERVER statistics" on page 576
JVM programs	"JVM program statistics" on page 583
LIBRARY	"LIBRARY statistics" on page 588
Loader domain	"Loader domain statistics" on page 597
Logstream	"Logstream statistics" on page 611
LSRpool	"LSR pool statistics" on page 617
Monitoring	"Monitoring domain statistics" on page 632
MQCONN	"WebSphere MQ Connection statistics" on page 786
PIPELINE definitions	"PIPELINE definition statistics" on page 645
Program	"Program statistics" on page 649
Program autoinstall	"Program autoinstall statistics" on page 644
Program definitions	"Program definition statistics" on page 657
Recovery manager	"Recovery manager statistics" on page 667
Statistics domain	"Statistics domain statistics" on page 678
Storage manager	"Storage manager statistics" on page 681
Table manager	"Table manager statistics" on page 700
TCP/IP	"TCP/IP global and TCP/IP Service statistics" on page 701
Temporary storage	"Temporary storage statistics" on page 712
Terminal control	"Terminal control statistics" on page 722
Transaction class (TCLASS)	"Transaction class (TCLASS) statistics" on page 725
Transaction manager	"Transaction statistics" on page 731
Transient data	"Transient data statistics" on page 744
URIMAP definitions	"URIMAP definition statistics" on page 762
User domain	"User domain statistics" on page 775
Web services	"Web service statistics" on page 782

Table 36. CICS statistics areas (continued)

Statistic type	Topic
WebSphere MQ connection	"WebSphere MQ Connection statistics" on page 786
XMLTRANSFORM	"XMLTRANSFORM statistics" on page 793
z/OS Communications Server (VTAM)	"SNA statistics" on page 778

Server statistics not in DFHSTUP

The DFHSTUP summary report does not include the statistics obtained for the shared temporary storage queue server, the coupling facility data tables server, and the named counter sequence number server.

Shared temporary storage queue server statistics

Shared temporary storage queue server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

For more information, see "Shared temporary storage queue server statistics" on page 673.

Coupling facility data tables server statistics

Coupling facility data tables server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

For more information, see "Coupling facility data tables server statistics" on page 459.

Named counter sequence number server statistics

Named counter sequence number server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

For more information, see "Named counter sequence number server" on page 642.

The sample statistics program, DFH0STAT

The sample statistics program, DFH0STAT, produces a report showing system information about CICS resources and an overview of the MVS storage in use.

The program demonstrates how you can use **EXEC CICS INQUIRE**, **EXEC CICS COLLECT STATISTICS**, and **EXEC CICS EXTRACT STATISTICS** commands to produce an analysis of your CICS regions. You can use the sample program as supplied, or modify it to suit your needs.

DFH0STAT does not report on terminals, DBCTL resources, FEPI resources, dumps, the table manager, and the user domain. DFH0STAT also does not report on private resources for applications that are deployed on platforms. If you require statistical information about any of these areas, you can obtain it using DFHSTUP, the statistics utility program (see Statistics utility program (DFHSTUP) in the *CICS Operations and Utilities Guide*).

DFH0STAT does not always report to the maximum capacity of certain large statistics fields. If your CICS system is unusually large or very busy, and you have a long statistics interval, check that the statistics values have not overflowed. To avoid this problem, reduce the length of your statistics interval, or use DFHSTUP.

See "Information on DFH0STAT" for more information on the DFH0STAT sample program.

See Chapter 33, "DFH0STAT reports," on page 797 for a listing of DFH0STAT reports.

Information on DFH0STAT

The main programs for the sample statistics program DFH0STAT are written in COBOL and supplied in source form in the CICSTS53.CICS.SDFHSAMP library. DFH0STAT is also supplied in pregenerated form in CICSTS53.CICS.SDFHLOAD.

HTML versions of the BMS maps are supplied with the sample application, so that you can run the STAT transaction using CICS Web support.

The components of DFH0STAT are all defined in the CSD group DFH$STAT. They include a number of COBOL modules and some additional components such as map sets. The DFH$STAT CSD group also defines programs DFH$STED and DFH$STER, but these are not part of the DFH0STAT sample application.

The following COBOL modules are components of the sample statistics program DFH0STAT:

DFH0STAT
This is the main COBOL program, which handles all BMS screen input/output, and the open and close of the JES SPOOL. It links to DFH0STLK, which controls all the other routines.

DFH0STLK
This COBOL module is called from DFH0STAT. DFH0STLK performs the following functions:

- Initializes the page numbers
- Links to the other routines.
- Prints the page index if selected.

DFH0STDB
This COBOL module is called from DFH0STLK to print the collected statistics for:

- Files
- Data set names
- Data tables
- DB2 connection
- DB2 entries
- LSRpool
- WebSphere MQ connection

DFH0STEJ
This COBOL module is called from DFH0STLK to print the collected statistics for:

- JVMs
- JVM programs
- JVMSERVER resources

DFH0STEP
This COBOL module is called from DFH0STLK to print the collected statistics for:

- Event processing
- Event bindings
- Capture specifications

DFH0STGN
This COBOL module is called from DFH0STLK to print the collected statistics for:
- User exit programs
- Global user exits
- Trace settings and levels
- Enqueue manager
- Enqueue models
- Recovery manager

DFH0STPR
This COBOL module is called from DFH0STLK to print the collected statistics for:
- Journalnames
- Logstreams
- Program autoinstall
- Terminal autoinstall and z/OS Communications Server
- Connections and modenames
- TCP/IP
- TCP/IP services
- IPCONN resources

DFH0STSA
This COBOL module is called from DFH0STLK to print the collected statistics for:
- Storage analysis (DSAs)
- Loader
- LIBRARY resources
- LIBRARY data set concatenation

DFH0STSY
This COBOL module is called from DFH0STLK to print the collected statistics for:
- System Status
- Transaction manager
- Dispatcher
- Dispatcher MVS TCBs

DFH0STTP
This COBOL module is called from DFH0STLK to print the collected statistics for:
- Transaction classes
- Transactions
- Program definitions
- Programs (and programs by DSA and LPA)
- DFHRPL and LIBRARY analysis

DFH0STTS
This COBOL module is called from DFH0STLK to print the collected statistics for:
- Temporary storage
- Temporary storage main — storage subpools
- Temporary storage models
- Temporary storage queues
- Transient data

DFH0STWB
This COBOL module is called from DFH0STLK to print the collected statistics for:
- BUNDLE resources
- URIMAP resources
- Virtual hosts
- ATOMSERVICE resources
- PIPELINE resources
- WEBSERVICE resources
- DOCTEMPLATE resources
- XMLTRANSFORM resources

The additional components for DFH0STAT that are defined in the CSD group DFH$STAT are:

DFH0STCM
The communications area (COMMAREA) used for communication between all the COBOL programs in the DFH0STAT suite.

DFH$STAS
The assembler language subroutine called by the COBOL modules DFH0STSA and DFH0STSY.

DFH$STCN
The assembler language subroutine called by the other COBOL modules in the DFH0STAT suite.

DFH$STTB
The assembler language table of global user exit names, loaded by the COBOL module DFH0STGN.

DFH0STM
This is the name of one of the map set source files supplied in SDFHSAMP, and also the name of one of the physical map sets, used by STAT transaction in program DFH0STAT, supplied in SDFHLOAD.

DFH0STS
This is the name of one of the map set source files supplied in SDFHSAMP, and also the name of one of the physical map sets, used by STAT transaction in program DFH0STAT, supplied in SDFHLOAD.

DFH0STMU
The HTML version of the map set DFH0STM, supplied in SDFHSAMP.

DFH0STSU
The HTML version of the map set DFH0STS, supplied in SDFHSAMP.

STAT The transaction that invokes DFH0STAT.

The sample program can be invoked as follows:
- As a program list table post-initialization (PLTPI) program, after the DFHDELIM statement.
- As a program list table shut-down (PLTSD) program, before the DFHDELIM statement.
- As a conversational transaction from a CICS terminal
- From a console
- As a started transaction using the EXEC CICS START command from a user-written application program
- By a distributed program link request to DFH0STAT from a user-written application program

To enable you to run the pregenerated sample statistics program from a CICS terminal, ensure SPOOL=YES is specified as a system initialization parameter for the CICS region. All the required executable code and map sets are supplied ready for use in CICSTS53.CICS.SDFHLOAD.

To customize the sample statistics application programs:
- You can use the pregenerated map sets. The following map objects are supplied:
  - Physical map sets, as load modules in CICSTS53.CICS.SDFHLOAD, which you can use unchanged.
  - Symbolic map sets, named DFH0STMD and DFH0STSD, for use as COBOL copybooks in DFH0STAT to enable you to recompile the sample program. These are supplied in CICSTS53.CICS.SDFHSAMP.
  - Map set source macros DFH0STM and DFH0STS, in CICSTS53.CICS.SDFHSAMP, that you can modify if you decide to customize the maps as well as the sample application programs.
  - HTML versions of the maps to enable you to run the sample application using the CICS Web interface. For information on how to create and load the HTML versions of the maps into a template data set, see Creating the CICS data sets in the *CICS Installation Guide*. See also the sample data set creation job, DFHDEFDS, supplied in SDFHINST.
- If your COBOL compiler does not have the integrated CICS translator, first translate the customized COBOL program source code, using the translator options COBOL3 and SP.
- Compile the translated output to produce object code.
- 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 startup job stream.

Chapter 32. DFHSTUP reports

This section lists the CICS statistics and associated DFHSTUP reports, grouped by the type of statistics, and provides more information about interpreting the statistics.

Atom feed statistics

The W2 domain collects statistics for ATOMSERVICE resource definitions, which define Atom feeds.

Related reference:

"ATOMSERVICEs report" on page 797
The ATOMSERVICEs report shows information and statistics about ATOMSERVICE resource definitions, which define Atom feeds. This report is produced using a combination of **EXEC CICS INQUIRE ATOMSERVICE** and **EXEC CICS EXTRACT STATISTICS ATOMSERVICE** commands.

Atom feeds: Resource statistics

You can retrieve Atom feed statistics by using the **EXEC CICS EXTRACT STATISTICS ATOMSERVICE()** system command. They are mapped by the DFHW2RDS DSECT.

Table 37. Atom feeds: resource statistics

DFHSTUP name	Field name	Description
ATOMSERVICE Name	W2R_ATOMSERV_NAME	The name of the ATOMSERVICE resource definition. Reset characteristic: not reset
Atom document type	W2R_ATOMSERV_TYPE	The type of Atom document that is returned for this ATOMSERVICE resource definition. **Category** An Atom category document, which lists the categories for entries in a collection. **Collection** An Atom collection document, which contains a group of entry documents that can be edited. **Feed** An Atom feed document, which describes the metadata for a feed, and contains entry documents that provide data for the feed. **Service** An Atom service document, which provides information about the collections that are available on the server. Reset characteristic: not reset

Table 37. Atom feeds: resource statistics (continued)

DFHSTUP name	Field name	Description
Atom binding file	W2R_ATOMSERV_BINDING_FILE	The name of the Atom binding file for the resource used for the Atom feed. Reset characteristic: not reset
Atom configuration file	W2R_ATOMSERV_CONFIG_FILE	The name of the Atom configuration file containing the XML for the Atom document. Reset characteristic: not reset
Resource type for Atom feed	W2R_ATOMSERV_RESTYPE	The type of resource that provides the data for this Atom feed. **File** A CICS file. **Program** A service routine, which is a CICS application program written to supply content for Atom entries. **Tsqueue** A temporary storage queue. Reset characteristic: not reset
Resource name for Atom feed	W2R_ATOMSERV_RESNAME	The name of the CICS resource that provides the data for this Atom feed or collection. Reset characteristic: not reset
ATOMSERVICE reference count	W2R_ATOMSERV_REF_COUNT	The number of times this ATOMSERVICE resource definition was referenced. Reset characteristic: reset to zero
ATOMSERVICE referenced - disabled	W2R_ATOMSERV_REF_DISABLED	The number of times this ATOMSERVICE resource definition was referenced, but the resource definition was disabled. Reset characteristic: reset to zero
POST requests to the feed URL	W2R_ATOMSERV_POST_FEED_CNT	The number of HTTP POST requests to add a new Atom entry to this Atom feed or collection. Reset characteristic: reset to zero

Table 37. Atom feeds: resource statistics (continued)

DFHSTUP name	Field name	Description
GET requests to the feed URL	W2R_ATOMSERV_GET_FEED_CNT	The number of HTTP GET requests to obtain a group of entries from this Atom feed or collection. Reset characteristic: reset to zero
GET requests to the entry URL	W2R_ATOMSERV_GET_ENTRY_CNT	The number of HTTP GET requests to obtain an individual Atom entry from this Atom feed or collection. Reset characteristic: reset to zero
PUT requests to the entry URL	W2R_ATOMSERV_PUT_ENTRY_CNT	The number of HTTP PUT requests to edit an Atom entry in this Atom feed or collection. Reset characteristic: reset to zero
DELETE requests to the entry URL	W2R_ATOMSERV_DEL_ENTRY_CNT	The number of HTTP DELETE requests to delete an individual Atom entry from this Atom feed or collection. Reset characteristic: reset to zero
Not in DFHSTUP report	W2R_ATOMSERV_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset

Table 37. Atom feeds: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	W2R_ATOMSERV_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_ URIMAP	The name of the URIMAP resource that indicates the URI that is associated with this ATOMSERVICE resource. Reset characteristic: not reset
Not in DFHSTUP report	W2R_ATOMSERV_ XMLTRANSFORM	The name of the XMLTRANSFORM resource that is associated with this ATOMSERVICE resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Atom feeds: Summary resource statistics

Summary statistics are not available online.

Table 38. Atom feeds: Summary resource statistics

DFHSTUP name	Description
ATOMSERVICE Name	The name of the ATOMSERVICE resource definition.
Atom document type	The type of Atom document that is returned for this ATOMSERVICE resource definition. **Category** An Atom category document, which lists the categories for entries in a collection. **Collection** An Atom collection document, which contains a group of entry documents that can be edited. **Feed** An Atom feed document, which describes the metadata for a feed, and contains entry documents that provide data for the feed. **Service** An Atom service document, which provides information about the collections that are available on the server.
Atom binding file	The name of the Atom binding file for the resource used for the Atom feed.
Atom configuration file	The name of the Atom configuration file containing the XML for the Atom document.
Resource type for Atom feed	The type of resource that provides the data for this Atom feed. **File** A CICS file. **Program** A service routine, which is a CICS application program written to supply content for Atom entries. **Tsqueue** A temporary storage queue.
Resource name for Atom feed	The name of the CICS resource that provides the data for this Atom feed or collection.
ATOMSERVICE reference count	The number of times this ATOMSERVICE resource definition was referenced.
ATOMSERVICE referenced - disabled	The number of times this ATOMSERVICE resource definition was referenced, but the resource definition was disabled.

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.

Related reference:

"Program Autoinstall report" on page 862
The Program Autoinstall report shows information and statistics about the status of program autoinstall, catalog program definitions, and the number of autoinstalls that were attempted, rejected, and failed.

Autoinstall: Global statistics - Local definition

You can retrieve autoinstall global statistics by using the **EXEC CICS COLLECT STATISTICS AUTOINSTALL** system command. They are mapped by the DFHA04DS DSECT.

Table 39. Autoinstall: Global statistics - Local definition

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 z/OS Communications Server 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

Table 39. Autoinstall: Global statistics - Local definition (continued)

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 z/OS Communications Server 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

Related reference:

COLLECT STATISTICS in Reference > System programming

Autoinstall: Global statistics - Remote definitions - shipped terminal statistics

Statistics related to remote and shipped terminal definitions.

Table 40. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics

DFHSTUP name	Field name	Description
Delete shipped 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

Table 40. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
Delete shipped 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. (which equates to the sum of "Shipped terminals installed" and "Shipped terminals timed out"). 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 timed out	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 delete shipped 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
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

Table 40. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
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	is the current number of remote terminal definitions that are idle and are awaiting reuse. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	A04CIDLE	is the total time that the current number of remote terminal definitions that are awaiting reuse have been idle. Reset characteristic: Not reset

Table 40. Autoinstall: Global statistics - Remote definitions - shipped terminal statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A04CMAXI	is the current maximum time that a remote terminal definition that is awaiting reuse has been idle. Reset characteristic: Not reset

Autoinstall: Summary global statistics

Autoinstall summary global statistics are not available online.

Table 41. Autoinstall: Summary global statistics

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 z/OS Communications Server 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 z/OS Communications Server 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.
Delete shipped 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.
Delete shipped 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 (which equates to the sum of “Shipped terminals installed”, a statistic not shown in the summary report, and “Shipped terminals timed out”).
Shipped terminals timed out	is the number of shipped remote terminal definitions deleted during the recording period by the TIMEOUT transaction.

Table 41. Autoinstall: Summary global statistics (continued)

DFHSTUP name	Description
Times interval expired	is the number of times the delete shipped interval 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 during the recording period.
Remote deletes issued	is the number of old-style (pre-CICS/ESA 4.1) remote delete instructions issued by this region during the recording period.
Successful remote deletes	is the number of shipped terminal definitions deleted from this region because of old-style remote delete instructions, during 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).
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).

BUNDLE statistics

The resource life-cycle (RL) domain collects statistics for BUNDLE resource definitions, which define application bundles in a CICS region.

Related reference:

"Bundles Report" on page 798
The Bundles Report shows information and statistics about BUNDLE resources. The BUNDLE resource defines where a CICS bundle is deployed on z/OS UNIX and its status.

Bundles: resource statistics

You can retrieve bundle statistics by using the **EXEC CICS EXTRACT STATISTICS BUNDLE** system command. They are mapped by the DFHRLRDS DSECT.

Table 42. Bundles: resource statistics

DFHSTUP name	Field name	Description
Bundle name	RLR_BUNDLE_NAME	The name of the BUNDLE resource definition. Reset characteristic: not reset
Bundle directory	RLR_BUNDLE_DIRECTORY	The location of the bundle on z/OS UNIX. Reset characteristic: not reset

Table 42. Bundles: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	RLR_BUNDLE_BASESCOPE	The scope that is associated with the BUNDLE resource. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	RLR_BUNDLE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

BUNDLE: Summary resource statistics

Summary statistics are not available online.

Table 43. Bundles: Summary resource statistics

DFHSTUP name	Description
BUNDLE name	The name of the BUNDLE resource definition.
BUNDLE directory	The location of the bundle in z/OS UNIX.

CICS DB2 statistics

Related concepts:

"Interpreting CICS DB2 statistics"

Related reference:

"DB2 Connection report" on page 806
The DB2 Connection report shows information and statistics about DB2 Connection resource definitions, which define the connection between CICS and DB2 for a CICS region. The report also includes statistics about pool threads, DSNC commands, and tasks that wait for a TCB or pool thread.

"DB2 Entries report" on page 811
The DB2 Entries Report is produced using a combination of the **EXEC CICS INQUIRE DB2ENTRY** and **EXEC CICS EXTRACT STATISTICS DB2ENTRY** commands. The statistics data is mapped by the **DFHD2RDS DSECT**.

Interpreting CICS DB2 statistics

In addition to the limited statistics output by the **DSNC DISP STAT** command and those output to the STATSQUEUE destination of the DB2CONN during attachment facility shutdown, a more comprehensive set of CICS DB2 statistics can be collected using standard CICS statistics interfaces:

- The **EXEC CICS EXTRACT STATISTICS** command accepts the DB2CONN keyword to allow CICS DB2 global statistics to be collected. CICS DB2 global statistics are mapped by the DFHD2GDS DSECT.
- The **EXEC CICS EXTRACT STATISTICS** command accepts the DB2ENTRY() keyword to allow CICS DB2 resource statistics to be collected for a particular DB2ENTRY. CICS DB2 resource statistics are mapped by the DFHD2RDS DSECT.
- The **EXEC CICS PERFORM STATISTICS** command accepts the DB2 keyword to allow the user to request that CICS DB2 global and resource statistics are written out to SMF.

The CICS DB2 global and resource statistics are described in the CICS statistics tables, "CICS DB2 statistics." For more information about CICS and DB2, see Overview of the CICS DB2 interfacein the *CICS DB2 Guide*. Chapter 15, "Database management for performance," on page 219 deals with CICS DB2 performance.

CICS DB2: Global statistics

You can retrieve CICS DB2 global statistics by using the **EXEC CICS EXTRACT STATISTICS DB2CONN** system command. They are mapped by the DFHD2GDS DSECT.

Table 44. CICS DB2: Global statistics

DFHSTUP name	Field name	Description
DB2 Connection Name	D2G_DB2CONN_NAME	The name of the installed DB2CONN. Reset characteristic: not reset
DB2 Groupid	D2G_DB2_GROUP_ID	The name of a data-sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group. If CICS is connected to DB2, or is waiting to reconnect to a specific DB2 subsystem to resynchronize outstanding units of work, D2G_DB2_ID shows the member of the data-sharing group that has been chosen. Reset characteristic: not reset
Resync Group Member	D2G_RESYNCMEMBER	The action CICS takes if you are using group attach, with a DB2 group ID (D2G_DB2_GROUP_ID) set, and outstanding units of work are being held for the last DB2 data sharing group member to which CICS was connected. `Yes` means that CICS reconnects to the last connected DB2 data sharing group member. `No` means that CICS makes one attempt to reconnect to the last connected DB2 data sharing group member, and if that attempt fails, it connects to any member of the DB2 data sharing group. If you are not using group attach, this DSECT field contains nulls (which are shown as N/A in the reports). Reset characteristic: not reset
DB2 Sysid	D2G_DB2_ID	The name of the DB2 subsystem that CICS is connected to, or if a DB2 subsystem ID is specified in the installed DB2CONN definition, the DB2 subsystem that CICS connects to. If a DB2 group ID (D2G_DB2_GROUP_ID) is specified in the installed DB2CONN definition instead of a DB2 subsystem ID, and CICS is not currently connected to DB2, D2G_DB2_ID is normally blank. However, if a DB2 group ID is specified, but CICS is waiting to reconnect to a specific DB2 subsystem to resynchronize outstanding units of work, D2G_DB2_ID shows the ID of the DB2 subsystem to which CICS is waiting to reconnect. Reset characteristic: not reset

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
DB2 Connect Date / Time	D2G_CONNECT_TIME_LOCAL	The local time when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value. Reset characteristic: not reset
DB2 Disconnect Date / Time	D2G_DISCONNECT_TIME_ LOCAL	The local time when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a local store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shut down, after which the time field is cleared to nulls (which are shown as N/A in the reports). Reset characteristic: not reset
DB2 Release	D2G_DB2_RELEASE	The version and release level of the DB2 subsystem that CICS is connected to. If CICS is not currently connected to DB2 the DSECT field contain nulls (which are shown as N/A in the reports). Reset characteristic: not reset
TCB Limit	D2G_TCB_LIMIT	The maximum number of TCBs that can be used by the CICS-DB2 attachment facility. Reset characteristic: not reset
Current number of Connections	D2G_TCB_CURRENT	The current number of connections associated with OPEN TCBs used by the CICS-DB2 attachment facility. Reset characteristic: not reset
Peak number of Connections	D2G_TCB_HWM	The peak number of connections associated with OPEN TCBs used by the CICS-DB2 attachment facility. Reset characteristic: reset to current value (D2G_TCB_CURRENT)

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Current number of free Connections	D2G_TCB_FREE	The number of free connections available for use with CICS open TCBs. Reset characteristic: not reset
Current number of tasks on the TCB Readyq	D2G_TCB_READYQ_CURRENT	The number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Reset characteristic: not reset
Peak number of tasks on the TCB Readyq	D2G_TCB_READYQ_HWM	The peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Reset characteristic: reset to current value (D2G_TCB_READYQ_CURRENT)
Thread reuselimit	D2G_REUSELIMIT	The maximum number of times a thread can be reused before being terminated. Reset characteristic: not reset
Total times reuselimit hit by a pool thread	D2G_POOL_REUSELIMIT_COUNT	The number of times the reuselimit has been reached by a pool thread. Reset characteristic: reset to zero
Pool Thread Plan name	D2G_POOL_PLAN_NAME	The name of the plan used for the pool. If a dynamic plan exit is being used for the pool this DSECT field will be nulls (which are shown as N/A in the reports). Reset characteristic: not reset
Pool Thread Dynamic Planexit name	D2G_POOL_PLANEXIT_NAME	The name of the dynamic plan exit to be used for the pool. If a static plan is being used for the pool this DSECT field will be nulls (which are shown as N/A in the reports). Reset characteristic: not reset

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Pool Thread Authtype	D2G_POOL_AUTHTYPE	The type of id to be used for DB2 security checking for pool threads. If an Authid is being used for pool threads this DSECT field contains nulls (which are shown as N/A in the reports). Reset characteristic: not reset
Pool Thread Authid	D2G_POOL_AUTHID	The static id to be used for DB2 security checking for pool threads. If an Authtype is being used for pool threads this DSECT field contains nulls (which are shown as N/A in the reports). Reset characteristic: not reset
Pool Thread Accountrec setting	D2G_POOL_ACCOUNTREC	Specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. Reset characteristic: not reset
Pool Thread Threadwait setting	D2G_POOL_THREADWAIT	Specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads exceed the pool thread limit. Reset characteristic: not reset
Pool Thread Priority	D2G_POOL_PRIORITY	The priority of the pool thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing not applicable (which is shown as N/A in the reports). Reset characteristic: not reset
Number of calls using Pool Threads	D2G_POOL_CALLS	The number of SQL calls made using pool threads. If you are using PACKAGESET support for cloud applications, this number includes the number of **EXEC SQL SET CURRENT PACKAGESET** commands issued by DFHD2SPS on behalf of the application. Reset characteristic: reset to zero
Number of Pool Thread Signons	D2G_POOL_SIGNONS	The number of DB2 signons performed for pool threads. Reset characteristic: reset to zero

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of Pool Thread Partial Signons	D2G_POOL_PARTIAL_SIGNONS	The number of DB2 partial signons performed for pool threads. Reset characteristic: reset to zero
Number of Pool Thread Commits	D2G_POOL_COMMITS	The number of 2 phase commits performed for units of work using pool threads. Reset characteristic: reset to zero
Number of Pool Thread Aborts	D2G_POOL_ABORTS	The number of units of work using pool threads that were rolled back. Reset characteristic: reset to zero
Number of Pool Thread Single Phases	D2G_POOL_SINGLE_PHASE	The number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Reset characteristic: reset to zero
Number of Pool Thread Creates	D2G_POOL_THREAD_CREATE	The number of times that CICS transactions using the pool create a DB2 thread. This count includes transactions that overflow to the pool to acquire a thread. Reset characteristic: reset to zero
Number of Pool Thread Reuses	D2G_POOL_THREAD_REUSE	The number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. Reset characteristic: reset to zero
Number of Pool Thread Terminates	D2G_POOL_THREAD_TERM	The number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. Reset characteristic: reset to zero

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of Pool Thread Waits	D2G_POOL_THREAD_WAITS	The number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and must wait for a pool thread. Reset characteristic: reset to zero
Current Pool Thread Limit	D2G_POOL_THREAD_LIMIT	The current maximum number of pool threads allowed. Reset characteristic: not reset
Current number of Pool Threads in use	D2G_POOL_THREAD_CURRENT	The current number of active pool threads. Reset characteristic: not reset
Peak number of Pool Threads in use	D2G_POOL_THREAD_HWM	The peak number of active pool threads. Reset characteristic: reset to current value (D2G_POOL_THREAD_CURRENT)
Current number of Pool tasks	D2G_POOL_TASK_CURRENT	The current number of CICS tasks that are using a pool thread. Reset characteristic: not reset
Peak number of Pool tasks	D2G_POOL_TASK_HWM	The peak number of CICS tasks that have used a pool thread. Reset characteristic: reset to current value (D2G_POOL_TASK_CURRENT)
Total number of Pool tasks	D2G_POOL_TASK_TOTAL	The total number of completed tasks that have used a pool thread. Reset characteristic: reset to zero.
Current number of tasks on the Pool Readyq	D2G_POOL_READYQ_CURRENT	The current number of CICS tasks waiting for a pool thread to become available. Reset characteristic: not reset

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Peak number of tasks on the Pool Readyq	D2G_POOL_READYQ_HWM	The peak number of CICS tasks that waited for a pool thread to become available. Reset characteristic: reset to current value (D2G_POOL_READYQ_CURRENT)
Command Thread Authtype	D2G_COMD_AUTHTYPE	The type of id to be used for DB2 security checking for command threads. If an Authid is being used for command threads this DSECT field contains nulls (which are shown as N/A in the reports). Reset characteristic: not reset
Command Thread Authid	D2G_COMD_AUTHID	The static id to be used for DB2 security checking for command threads. If an Authtype is being used for command threads this DSECT field contains nulls (which are shown as N/A in the reports). Reset characteristic: not reset
Number of calls using Command Threads	D2G_COMD_CALLS	The number of DB2 commands issued using the DSNC transaction. Reset characteristic: reset to zero
Number of Command Thread Signons	D2G_COMD_SIGNONS	The number of DB2 signons performed for command threads. Reset characteristic: reset to zero
Number of Command Thread Creates	D2G_COMD_THREAD_CREATE	The number of create thread requests made to DB2 for command threads. Reset characteristic: reset to zero
Number of Command Thread Terminates	D2G_COMD_THREAD_TERM	The number of terminate thread requests made to DB2 for command threads. Reset characteristic: reset to zero
Number of Command Thread Overflows to Pool	D2G_COMD_THREAD_OVERF	The number of times a DSNC DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit. Reset characteristic: reset to zero

Table 44. CICS DB2: Global statistics (continued)

DFHSTUP name	Field name	Description
Command Thread Limit	D2G_COMD_THREAD_LIMIT	The current maximum number of command threads allowed. Reset characteristic: not reset
Current number of Command Threads	D2G_COMD_THREAD_CURRENT	The current number of active command threads. Reset characteristic: not reset
Peak number of Command Threads	D2G_COMD_THREAD_HWM	The peak number of active command threads. Reset characteristic: reset to current value (D2G_COMD_THREAD_CURRENT)
NOT IN THE DFHSTUP REPORT	D2G_CONNECT_TIME_GMT	The Greenwich mean time (GMT) when CICS connected to DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	D2G_DISCONNECT_TIME_GMT	The Greenwich mean time (GMT) when CICS disconnected from DB2. The DFHSTUP report expresses this time as hh:mm:ss; however the DSECT field contains the time as a GMT store clock (STCK) value. The disconnect time will only be present in DB2CONN unsolicited statistics records produced when the CICS DB2 interface is shut down, after which the time field is cleared to nulls (which are shown as N/A in the reports). Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

CICS DB2: Resource statistics

You can retrieve CICS DB2 resource statistics by using the **EXEC CICS EXTRACT STATISTICS DB2ENTRY** system command. They are mapped by the DFHD2RDS DSECT.

Related information:

EXTRACT STATISTICS in Reference > System programming

CICS DB2: Resource statistics - resource information

The resource information gives details of various attribute settings of each DB2ENTRY resource.

Table 45. CICS DB2 : Resource statistics - resource information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	The name of the installed DB2ENTRY Reset characteristic: not reset
Plan Name	D2R_PLAN_NAME	The name of the plan used for this DB2ENTRY. If a dynamic plan exit is used for the DB2Entry, this DSECT field will be nulls (which are shown as N/A in the reports). Reset characteristic: not reset
PlanExit name	D2R_PLANEXIT_NAME	The name of the dynamic plan exit to be used for this DB2ENTRY. If a static plan is used for the DB2ENTRY this DSECT field is nulls, which are shown as N/A in the reports. Reset characteristic: not reset
Auth Id	D2R_AUTHID	The static ID to be used for DB2 security checking for this DB2ENTRY. If an Authtype is used for the DB2ENTRY this DSECT field is nulls, which are shown as N/A in the reports. Reset characteristic: not reset
Auth Type	D2R_AUTHTYPE	The type of ID to be used for DB2 security checking for this DB2ENTRY. If an Authid is used for the DB2ENTRY this DSECT field contains nulls, which are shown as N/A in the reports. Reset characteristic: not reset
Account Records	D2R_ACCOUNTREC	Specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY. Reset characteristic: not reset
Thread Wait	D2R_THREADWAIT	Specifies whether transactions wait for a thread, stop or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit. Reset characteristic: not reset
Thread Prty	D2R_PRIORITY	The priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing not applicable, which is shown as N/A in the reports. Reset characteristic: not reset

Table 45. CICS DB2 : Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	D2R_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	D2R_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	D2R_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	D2R_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	D2R_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	D2R_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	D2R_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

CICS DB2: Resource statistics - request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 46. CICS DB2: Resource statistics - request information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	is the name of the installed DB2ENTRY Reset characteristic: not reset
Call Count	D2R_CALLS	is the number of SQL calls made using this DB2ENTRY. If you are using PACKAGESET support for cloud applications, this number includes the number of **EXEC SQL SET CURRENT PACKAGESET** commands issued by DFHD2SPS on behalf of the application. Reset characteristic: reset to zero
Signon Count	D2R_SIGNONS	is the number of DB2 signons performed for this DB2ENTRY. Reset characteristic: reset to zero
Partial Signon	D2R_PARTIAL_SIGNONS	is the number of DB2 partial signons performed for this DB2ENTRY. Reset characteristic: reset to zero
Commit Count	D2R_COMMITS	is the number of two phase commits performed for units of work using this DB2ENTRY. Reset characteristic: reset to zero
Abort Count	D2R_ABORTS	is the number of units of work using this DB2ENTRY that were rolled back. Reset characteristic: reset to zero
Single Phase	D2R_SINGLE_PHASE	is the number of units of work using the DB2ENTRY that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Reset characteristic: reset to zero
Thread Create	D2R_THREAD_CREATE	is the number of create thread requests made to DB2 for threads of this DB2ENTRY. Reset characteristic: reset to zero

Table 46. CICS DB2: Resource statistics - request information (continued)

DFHSTUP name	Field name	Description
Thread Reuse	D2R_THREAD_REUSE	is the number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread. Reset characteristic: reset to zero
Thread Terms	D2R_THREAD_TERM	is the number of terminate thread requests made to DB2 for threads of this DB2ENTRY. Reset characteristic: reset to zero
Thread Waits/Overflows	D2R_THREAD_WAIT_ OR_OVERF	is the number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead. Reset characteristic: reset to zero

CICS DB2: Resource statistics - performance information

The performance information gives details of Thread information for each DB2ENTRY.

Table 47. CICS DB2: Resource statistics - performance information

DFHSTUP name	Field name	Description
DB2Entry Name	D2R_DB2ENTRY_NAME	The name of the installed DB2ENTRY Reset characteristic: not reset
Thread Limit	D2R_THREAD_LIMIT	The current maximum number of threads allowed for the DB2ENTRY. Reset characteristic: not reset
Thread Current	D2R_THREAD_CURRENT	The current number of active threads for this DB2ENTRY. Reset characteristic: not reset
Thread HWM	D2R_THREAD_HWM	The peak number of active threads for this DB2ENTRY. Reset characteristic: reset to current value (D2R_THREAD_CURRENT)

Table 47. CICS DB2: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Pthread Limit	D2R_PTHREAD_LIMIT	The current maximum number of protected threads allowed for this DB2ENTRY. Reset characteristic: not reset
Pthread Current	D2R_PTHREAD_CURRENT	The current number of protected threads for this DB2ENTRY. Reset characteristic: not reset
Pthread HWM	D2R_PTHREAD_HWM	The peak number of protected threads for this DB2ENTRY. Reset characteristic: reset to current value (D2R_PTHREAD_CURRENT)
Task Current	D2R_TASK_CURRENT	The current number of CICS tasks that are using this DB2ENTRY. Reset characteristic: not reset
Task HWM	D2R_TASK_HWM	The peak number of CICS tasks that have used this DB2ENTRY. Reset characteristic: reset to current value (D2R_TASK_CURRENT)
Task Total	D2R_TASK_TOTAL	The total number of completed tasks that have used this DB2ENTRY. Reset characteristic: reset to zero.
Readyq Current	D2R_READYQ_CURRENT	The current number of CICS tasks waiting for a thread to become available on this DB2ENTRY. Reset characteristic: not reset
Readyq HWM	D2R_READYQ_HWM	The peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY. Reset characteristic: reset to current value (D2R_READYQ_CURRENT)
Reuselm hits	D2R_REUSELIMIT_COUNT	The number of times the reuselimit has been reached by a thread for this DB2ENTRY. Reset characteristic: reset to zero.

CICS DB2: Summary global statistics

Shows summary information and statistics about CICS DB2. Summary statistics are unavailable online.

Table 48. CICS DB2: Summary global statistics

DFHSTUP name	Description
DB2 Connection Name	The name of the installed DB2CONN.
Total DB2 Connection time	The total amount of time CICS was connected to the DB2 subsystem specified in this DB2CONN. The time is displayed as days:hh:mm:ss.
DB2 Groupid	The name of a data sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group.
Resync Group Member	Specifies the action CICS takes if you are using group attach, with a DB2 group ID set, and outstanding units of work are being held for the last DB2 data sharing group member to which CICS was connected. 'Yes' means that CICS reconnects to the last connected DB2 data sharing group member. 'No' means that CICS makes one attempt to reconnect to the last connected DB2 data sharing group member, and if that attempt fails, it connects to any member of the DB2 data sharing group. If you are not using group attach, N/A is shown in the report.
DB2 Sysid	The name of the DB2 subsystem to which CICS connects, as specified in the installed DB2CONN definition. If the sysid has changed, it is the last setting of sysid.
DB2 Release	The DB2 version and release for this DB2CONN. If the version and release have changed, it is the last setting of version and release.
TCB Limit	The TCBLIMIT value that was set in the DB2CONN. If the TCBLIMIT has changed, it is the last setting of TCBLIMIT. The TCB limit is the maximum number of TCBs that can be used by the CICS-DB2 attachment facility.
Current number of Connections	The current number of connections used by the CICS-DB2 attachment facility.
Peak number of Connections	The peak number of connections used by the CICS-DB2 attachment facility.
Peak number of tasks on the TCB Readyq	The peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached.
Pool Thread Plan name	The name of the plan used for the pool. If the plan name has changed, it is the last setting of plan name. If a dynamic plan exit is being used for the pool, the summary report shows 'N/A'.
Pool Thread Dynamic Planexit name	The name of the dynamic plan exit to be used for the pool. If the dynamic plan exit name has changed, it is the last setting of dynamic planexit name. If static plan is being used for the pool, the summary report shows 'N/A'.
Pool Thread Authtype	The type of id to be used for DB2 security checking for pool threads. If the pool thread authtype has changed, it is the last setting of pool thread authtype. If an Authid is being used for pool threads, the summary report shows 'N/A'.

Table 48. CICS DB2: Summary global statistics (continued)

DFHSTUP name	Description
Pool Thread Authid	The static id to be used for DB2 security checking for pool threads. If the pool thread authid has changed, it is the last setting of pool thread authid. If an Authtype is being used for pool threads, the summary report shows 'N/A'.
Pool Thread Accountrec setting	The frequency of DB2 accounting records to be produced for transactions using pool threads. If the pool thread accountrec setting has changed, it is the last setting of pool thread accountrec.
Pool Thread Threadwait setting	The setting for whether transactions should wait for a pool thread or be abended if the number of active pool threads reaches the pool thread limit. If the pool thread threadwait setting has changed, it is the last setting of pool thread threadwait.
Pool Thread Priority	The priority of the pool thread subtasks relative to the CICS main task (QR TCB). If the pool thread priority has changed, it is the last setting of pool thread priority. If CICS is connected to DB2 Version 6 or later, this field contains zero (representing not applicable), and the summary report shows 'N/A'.
Total number of calls using Pool Threads	The total number of SQL calls made using pool threads.
Total number of Pool Thread Signons	The total number of DB2 signons performed for pool threads.
Total number of Pool Thread Partial Signons	The total number of DB2 partial signons performed for pool threads.
Total number of Pool Thread Commits	The total number of two phase commits performed for units of work using pool threads.
Total number of Pool Thread Aborts	The total number of units of work using pool threads that were rolled back.
Total number of Pool Thread Single Phases	The total number of units of work using pool threads that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.
Total number of Pool Thread Reuses	The total number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread.
Total number of Pool Thread Terminates	The total number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool.
Total number of Pool Thread Waits	The total number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread.

Table 48. CICS DB2: Summary global statistics (continued)

DFHSTUP name	Description
Pool Thread Limit	The thread limit value for the pool. If the pool thread limit has changed, it is the last setting of pool thread limit.
Peak number of Pool Threads in use	The peak number of active pool threads.
Peak number of Pool tasks	The peak number of CICS tasks that have used a pool thread.
Total number of Pool tasks	The total number of completed tasks that have used a pool thread.
Peak number of tasks on the Pool Readyq	The peak number of CICS tasks that waited for a pool thread to become available.
Command Thread Authtype	The type of id to be used for DB2 security checking for command threads. If the command thread authtype has changed, it is the last setting of command thread authtype. If an Authid is being used for command threads, the summary report shows 'N/A'.
Command Thread Authid	The static id to be used for DB2 security checking for command threads. If the command thread authid has changed, it is the last setting of command thread authid. If an Authtype is being used for command threads, the summary report shows 'N/A'.
Total number of Command Thread Calls	The total number of DB2 commands issued through the DSNC transaction.
Total number of Command Thread Signons	The total number of DB2 signons performed for command threads.
Total number of Command Thread Terminates	The total number of terminate thread requests made to DB2 for command threads.
Total number of Command Thread Overflows	The total number of times a DSNC DB2 command resulted in a pool thread being used because the number of active command threads exceed the command thread limit.
Command Thread Limit	The maximum number of command threads allowed. If the command thread limit has changed, it is the last setting of command thread limit.
Peak number of Command Threads	The peak number of active command threads.

CICS DB2: Summary resource statistics

The CICS DB2 resource statistics summary report DFHSTUP contains three sections: resource information, request information, and performance information.

Summary statistics are unavailable online.

CICS DB2: Summary resource statistics - resource information

The resource information gives details of various attribute settings of each DB2ENTRY.

Table 49. CICS DB2: Summary resource statistics - resource information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY.
Plan Name	is the name of the plan used for this DB2ENTRY. If the plan name changed, it is the last setting of plan name. If a dynamic plan exit is being used for the DB2Entry, the summary report shows 'N/A'.
PlanExit Name	is the name of the dynamic plan exit to be used for this DB2ENTRY. If the plan exit name has changed, it is the last setting of PlanExit name. If a static plan is being used for the DB2ENTRY, the summary report shows 'N/A'.
Auth Id	is the static id to be used for DB2 security checking for this DB2ENTRY. If the Auth id changed, it is the last setting of Auth id. If an Authtype is being used for the DB2ENTRY, the summary report shows 'N/A'.
Auth Type	is the type of id to be used for DB2 security checking for this DB2ENTRY. If the Auth type changed, it is the last setting of Auth type. If an Authid is being used for the DB2ENTRY, the summary report shows 'N/A'.
Account Records	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY. If the frequency changed, it is the last frequency setting.
Thread Wait	specifies whether transactions should wait for a thread, abend, or overflow to the pool, if the number of active threads for this DB2ENTRY exceeds its thread limit. If the threadwait changed, it is the last setting of threadwait.
Thread Prty	is the priority of the DB2ENTRY thread subtasks relative to the CICS main task (QR TCB). If the priority changed, it is the last setting of priority. If CICS is connected to DB2 Version 6 or later, this field contains zero (representing not applicable), and the summary report shows 'N/A'.

CICS DB2: Summary resource statistics - request information

The request information gives details of how many requests of various types have been performed against each DB2ENTRY.

Table 50. CICS DB2: Summary resource statistics - request information

DFHSTUP name	Description
DB2Entry Name	is the name of the installed DB2ENTRY.
Call Count	is the total number of SQL calls made using this DB2ENTRY.
Signon Count	is the total number of DB2 signons performed for this DB2ENTRY.
Partial Signon	is the total number of DB2 partial signons performed for this DB2ENTRY.
Commit Count	is the total number of two phase commits performed for units of work using this DB2ENTRY.
Abort Count	is the total number of units of work using this DB2ENTRY that were rolled back.
Single Phase	is the total number of units of work using the DB2ENTRY that used single phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW.

Table 50. CICS DB2: Summary resource statistics - request information (continued)

DFHSTUP name	Description
Thread Reuse	is the total number of times CICS transactions using the DB2ENTRY were able to reuse an already created DB2 thread.
Thread Terms	is the total number of terminate thread requests made to DB2 for threads of this DB2ENTRY.
Thread Waits/Overflows	is the total number of times all available threads in the DB2ENTRY were busy and a transaction had to wait for a thread to become available, or overflow to the pool and use a pool thread instead.

CICS DB2: Summary resource statistics - performance information

The performance information gives details of thread information for each DB2ENTRY.

Table 51. CICS DB2: Summary resource statistics - performance information

DFHSTUP name	Description
DB2ENTRY Name	is the name of the installed DB2ENTRY
Thread Limit	is the maximum number of threads allowed for the DB2ENTRY. If the value changed, it is the last setting of Thread limit.
Thread HWM	is the peak number of active threads for this DB2ENTRY.
Pthread Limit	is the maximum number of protected threads allowed for this DB2ENTRY. If the value changed, it is the last setting of Pthread limit.
Pthread HWM	is the peak number of protected threads for this DB2ENTRY.
Task HWM	is the peak number of CICS tasks that have used this DB2ENTRY.
Task Total	is the total number of completed tasks that have used this DB2ENTRY.
Readyq HWM	is the peak number of CICS tasks that waited for a thread to become available on this DB2ENTRY.

Coupling facility data tables server statistics

Coupling facility data tables server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

The CFDT statistics are calculated from information that is returned by recent coupling facility requests. If the relevant information was not accessed recently by the current server, the statistics are not necessarily accurate. The number of tables and the number of lists are updated each time the server opens or closes a table, but at other times they might not be updated. The element and entry counts are updated on successful completion of most types of coupling facility access request.

Related reference:

"Coupling Facility Data Table Pools report" on page 803
The Coupling Facility Data Table Pools report shows information and statistics about Coupling Facility Data Table Pools, which contain one or more coupling facility data tables.

Coupling facility data tables: list structure statistics

The statistics are described in detail in the DFHCFS6D data area.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The individual fields have the following meanings.

Table 52. Coupling facility data tables: list structure statistics

Statistic name	Field	Description
Structure		
	S6NAME	Full name of list structure
	S6PREF	First part of structure name
	S6POOL	Pool name part of structure name
	S6CNNAME	Name of connection to structure
	S6CNPREF	Prefix for connection name
	S6CNSYSN	Own MVS system name from CVTSNAME
Size	S6SIZE	Current allocated size of the list structure.
Max size	S6SIZEMX	Maximum size to which this structure could be altered.
Lists		
Total	S6HDRS	Maximum number of list headers in the structure.
Control	S6HDRSCT	Number of lists in use for control information.
Data	S6HDRSTD	Number of lists in use for table data.
Structure		
Elem size	S6ELEMLN	Data element size used for the structure.
	S6ELEMPW	Data element size as a power of 2
	S6ELEMRT	Element side of entry:element ratio
	S6ENTRRT	Entry side of entry:element ratio
Entries		
In use	S6ENTRCT	Number of entries currently in use.
Max used	S6ENTRHI	Maximum number in use (since last reset).
Min free	S6ENTRLO	Minumum number of free entries (since last reset).
Total	S6ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Elements		
In Use	S6ELEMCT	Number of elements currently in use.
Max Used	S6ELEMHI	Maximum number in use (since last reset).

Table 52. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Min Free	S6ELEMLO	Minimum number of free elements (since last reset)
Total	S6ELEMMX	Total data elements in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
List entry counts		
	S6USEVEC	Usage vector, five pairs of words
	S6USEDCT	Number of entries on used list
	S6USEDHI	Highest number of entries on used list
	S6FREECT	Number of entries on free list
	S6FREEHI	Highest number of entries on free list
	S6INDXCT	Number of entries in table index
	S6INDXHI	Highest entries in table index
	S6APPLCT	Number of entries in APPLID list
	S6APPLHI	Highest entries in APPLID list
	S6UOWLCT	Number of entries in UOW list
	S6UOWLHI	Highest entries in UOW list
Main type of CF request		
Table index lists		
Reads	S6RDICT	Number of table index reads.
Write	S6WRICT	Number of table index writes to create new tables.
Rewrite	S6RWICT	Number of table index writes to update table status.
Delete	S6DLICT	Number of table index deletes.
Data list controls		
Writes	S6CRLCT	Number of times a new data list was allocated.
Rewrites	S6MDLCT	Number of times data list controls were modified.
Deletes	S6DLLCT	Number of times a data list was deleted for reuse.
Table data record		
Reads	S6RDDCT	Number of data entry reads.
Writes	S6WRDCT	Number of data entry writes.
Rewrites	S6RWDCT	Number of data entry rewrites.
Deletes	S6DLDCT	Number of data entry deletes.
Data list controls		
Reads	S6INLCT	Inquire on data list
Lock release messages		
Reads	S6RDMCT	Number of lock release messages read by this server.
Writes	S6WRMCT	Number of lock release messages sent by this server.
UOW index list		
Reads	S6RDUCT	Number of UOW list reads.
Writes	S6WRUCT	Number of UOW list writes (usually at PREPARE)
Rewrites	S6RWUCT	Number of UOW list rewites (usually at COMMIT).
Deletes	S6DLUCT	Number of UOW list deletes (usually after COMMIT).
APPLID index lists		

Table 52. Coupling facility data tables: list structure statistics (continued)

Statistic name	Field	Description
Read	S6RDACT	Read APPLID entry
Write	S6WRACT	Write APPLID entry
Rewrite	S6RWACT	Rewrite APPLID entry
Delete	S6DLACT	Delete APPLID entry
Internal CF requests		
	S6RRLCT	Reread entry for full data length
Asynch	S6ASYCT	Number of requests for which completion was asynchronous.
IXLLIST completion		
Normal	S6RSP1CT	Number of normal responses.
Len err	S6RSP2CT	Entry data was larger than the inputbuffer length, which normally results in a retry with a larger buffer.
Not fnd	S6RSP3CT	The specified entry (table or item) was not found.
Vers chk	S6RSP4CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List chk	S6RSP5CT	A list authority comparison failed, mismatch caused by table status update
List full	S6RSP6CT	A table reached the maximum number of items causing the relevant list to be marked as full.
Str full	S6RSP7CT	The list structure became full.
I/O err	S6RSP8CT	Some other error code was returned by IXLLIST.

Coupling facility data tables: table accesses statistics

These statistics are described in detail in the DFHCFS7D data area.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The individual fields have the following meanings:

Table 53. Coupling facility data tables:queue pool statistics

Statistic name	Field	Description
Access		
	S7TABLE	Table name padded with spaces
Vector		
	S7STATS	Statistics vector
Table requests		
Open	S7OCOPEN	Number of successful OPEN requests for the table.
Close	S7OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S7OCSET	Number of times new table status was set.
Delete	S7OCDELE	Number of times the table of that name was deleted.
Stats	S7OCSTAT	Extract table statistics.
Record requests		

Table 53. Coupling facility data tables:queue pool statistics (continued)

Statistic name	Field	Description
Point	S7RQPOIN	Number of POINT requests.
Highest	S7RQHIGH	Number of requests for current highest key.
Read	S7RQREAD	Number of READ requests (including those for UPDATE)
Read del	S7RQRDDL	Number of combined READ and DELETE requests.
Unlock	S7RQUNLK	Number of UNLOCK requests.
Loads	S7RQLOAD	Number of records written by initial load requests.
Write	S7RQWRIT	Number of WRITE requests for new records.
Rewrite	S7RQREWR	Number of REWRITE requests.
Delete	S7RQDELE	Number of DELETE requests
Del Mult	S7RQDELM	Number of multiple (generic) delete requests.

Coupling facility data tables: request statistics

These statistics are described in detail in the DFHCFS8D data area.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The individual fields have the following meanings:

Table 54. Coupling facility data tables:request statistics

Statistic name	Field	Description
Vector		
	S8STATS	Statistics vector
Table		
Open	S8OCOPEN	Number of successful OPEN requests for the table
Close	S8OCCLOS	Number of successful CLOSE requests for the table.
Set Attr	S8OCSET	Number of times new table status was set.
Delete	S8OCDELE	Number of times the table of that name was deleted.
Stats	S8OCSTAT	Number of times table access statistics were extracted.
Record		
Point	S8RQPOIN	Number of POINT requests.
Highest	S8RQHIGH	Number of requests for current highest key
Read	S8RQREAD	Number of READ requests (including those for UPDATE)
Read Del	S8RQRDDL	Number of combined READ and DELETE requests
Unlock	S8RQUNLK	Number of UNLOCK requests.
Loads	S8RQLOAD	Number of records written by initial load requests.
Write	S8RQWRIT	Number of WRITE requests for new records
Rewrite	S8RQREWR	Number of REQRITE requests.
Delete	S8RQDELE	Number of DELETE requests.

Table 54. Coupling facility data tables:request statistics (continued)

Statistic name	Field	Description
Del Mult	S8RQDELM	Number of multiple (generic) delete requests
Table		
Inquire	S8IQINQU	Number of INQUIRE table requests.
UOW		
Prepare	S8SPPREP	Number of units of work prepared.
Retain	S8SPRETA	Number of units of work whose locks were retained.
Commit	S8SPCOMM	Number of units of work committed.
Backout	S8SPBACK	Number of units of work backed out.
Inquire	S8SPINQU	Number of units of work INQUIRE requests.
Restart	S8SPREST	Number of times recoverable connections were restarted.

Coupling facility data tables: storage statistics

These statistics are returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The statistics are described in detail in the DFHCFS9D data area.

Table 55. Coupling facility data tables: storage statistics

Statistic name	Field	Description
LOC=ANY storage pool statistics.		
Name	S9ANYNAM	Name of the storage pool AXMPGANY.
Size	S9ANYSIZ	Size of the storage pool area.
	S9ANYPTR	Address of storage pool area.
	S9ANYMX	Total pages in the storage pool.
In Use	S9ANYUS	Number of used pages in the pool.
Free	S9ANYFR	Number of free pages in the pool.
Min Free	S9ANYLO	Lowest free pages (since reset).
Gets	S9ANYRQG	Storage GET requests.
Frees	S9ANYRQF	Storage FREE requests.
Fails	S9ANYRQS	GETs which failed to obtain storage.
Retries	S9ANYRQC	Compress (defragmentation) attempts.
LOC=BELOW storage pool statistics.		

Table 55. Coupling facility data tables: storage statistics (continued)

Statistic name	Field	Description
Name	S9LOWNAM	Pool name AXMPGLOW.
Size	S9LOWSIZ	Size of storage pool area.
	S9LOWPTR	Address of storage pool area.
	S9LOWMX	Total pages in the storage pool.
In Use	S9LOWUS	Number of used pages in the storage pool.
Free	S9LOWFR	Number of free pages in the storage pool.
Min Free	S9LOWLO	Lowest free pages (since reset).
Gets	S9LOWRQG	Storage GET requests.
Frees	S9LOWRQF	Storage FREE requests.
Fails	S9LOWRQS	GETs which failed to obtain storage.
	S9LOWRQC	Compress (defragmentation) attempts.

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 Statistics, monitoring, and performance for DBCTLin the *CICS IMS Database Control Guide*.

DBCTL session termination: Global statistics

These statistics are mapped by the DFHDBUDS DSECT.

Table 56. DBCTL session termination: Global statistics

DFHSTUP name	Field name	Description
CICS DBCTL session number	STADSENO	The number of the CICS-DBCTL session, which is incremented every time you connect and disconnect. Reset characteristic: not reset
DBCTL identifier	STATDBID	The name of the DBCTL session. Reset characteristic: not reset

Table 56. DBCTL session termination: Global statistics (continued)

DFHSTUP name	Field name	Description
DBCTL RSE name	STARSEN	The name of the DBCTL recoverable service element (RSE). Reset characteristic: not reset
Time CICS connected to DBCTL	STALCTIM	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
Time CICS disconnected from DBCTL	STALDTIM	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	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	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	The minimum value specified in the DRA startup parameter table. Reset characteristic: not reset
Maximum number of threads	STAMATHD	The maximum value specified in the DRA startup parameter table. Reset characteristic: not reset
Times minimum threads hit	STANOMITHD	The number of times the CICS-DBCTL session "collapsed" threads down to the minimum thread value. Reset characteristic: not reset
Times maximum threads hit	STANOMATHD	The number of times the CICS-DBCTL session has hit the maximum thread value. Reset characteristic: not reset

Table 56. DBCTL session termination: Global statistics (continued)

DFHSTUP name	Field name	Description
Elapsed time at maximum threads	STAELMAX	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 thread TCBs	STAHIWAT	The highest number of thread TCBs created throughout the CICS-DBCTL session. Due to the asynchronous nature of TCB creation and deletion, it is possible for the number of TCBs to exceed the maximum number of threads, although the number of TCBs with an active thread will not exceed the maximum thread value. Reset characteristic: not reset
Successful PSB schedules	STAPSBSU	The number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB). Reset characteristic: not reset

DBCTL session termination: Summary global statistics

DBCTL session termination summary global statistics are not available online.

Table 57. DBCTL session termination: Summary global statistics

DFHSTUP name	Description
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.
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 "collapsed" threads down to 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 thread TCBs	is the highest number of thread TCBs created throughout the CICS-DBCTL session. Due to the asynchronous nature of TCB creation and deletion, it is possible for the number of TCBs to exceed the maximum number of threads, although the number of TCBs with an active thread will not exceed the maximum thread value.

Table 57. DBCTL session termination: Summary global statistics (continued)

DFHSTUP name	Description
Successful PSB schedules	is the total number of times the CICS-DBCTL session has successfully scheduled a program specification block (PSB).

Dispatcher domain statistics

Related concepts:

"Interpreting dispatcher statistics" on page 83
Use TCB dispatcher statistics and dispatcher TCB pool statistics to understand how the CICS dispatcher is performing.

Related reference:

"Dispatcher report" on page 814
The Dispatcher report is produced using a combination of the **EXEC CICS INQUIRE SYSTEM** and **EXEC CICS EXTRACT STATISTICS DISPATCHER** commands. The statistics data is mapped by the **DFHDSGDS DSECT**.

"Dispatcher TCB Modes report" on page 818
The Dispatcher TCB Modes report is produced using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** command. The statistics data is mapped by the **DFHDSGDS DSECT**.

"Dispatcher TCB Pools report" on page 821
The Dispatcher TCB Pools report is produced for each TCB pool. The example shows the OPEN TCB pool. This report is produced using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** command. The statistics data is mapped by the **DFHDSGDS DSECT**.

"Dispatcher MVS TCBs report" on page 815
The Dispatcher MVS TCBs report is produced using the **EXEC CICS EXTRACT STATISTICS MVSTCB**,**EXEC CICS EXTRACT STATISTICS DISPATCHER**, and **EXEC CICS INQUIRE MVSTCB** commands. The statistics data is mapped by the **DFHDSGDS**, **DFHDSTDS**, and **DFHDSRDS DSECT**.

Dispatcher domain: Global statistics

You can retrieve dispatcher domain global statistics by using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** system command. They are mapped by the DFHDSGDS DSECT.

Table 58. Dispatcher domain: Global statistics

DFHSTUP name	Field name	Description
Dispatcher Start Date and Time	DSGLSTRT	is the date and time at which the CICS dispatcher started. This value can be used as an approximate time at which CICS started. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. Reset characteristic: not reset

Table 58. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
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
Address Space CPU Time	DSGEJST	is the total CPU time for all TCBs in this address space, accumulated during the interval. The DFHSTUP report expresses this as *days-hours:minutes:seconds.decimals.* Reset characteristic: reset to zero
Address Space SRB Time	DSGSRBT	is the total CPU time for all service request blocks (SRB) executed in this address space, accumulated during the interval. The DFHSTUP report expresses this as *days-hours:minutes:seconds.decimals.* Reset characteristic: reset to zero
Current number of dispatcher tasks	DSGCNT	is the current number of dispatcher tasks in the system. This figure includes all system tasks and all user tasks. Reset characteristic: not reset
Peak number of dispatcher tasks	DSGPNT	is the peak value of the number of dispatcher 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 the **EXEC CICS SET SYSTEM TIME(fullword binary data-value)** command. Reset characteristic: not reset
Current ICVR time (msec)	DSGICVRT	is the ICVR time value (expressed im milliseconds) specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM TIME(fullword binary data-value)** command. Reset characteristic: not reset
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 the **EXEC CICS SET SYSTEM SCANDELAY(fullword binary data-value)** command. Reset characteristic: not reset

Table 58. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Current PRTYAGE time (msec)	DSGPRIAG	is the PRTYAGE time value (expressed in milliseconds) specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM AGING(fullword binary data-value)** command. Reset characteristic: not reset
Current MRO (QR) Batching (MROBTCH) value	DSGMBTCH	is the MROBTCH value specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM MROBATCH(fullword binary data-value)** command. Reset characteristic: not reset
Last Excess TCB Scan	DSGLXSCN	The date and time of the last CICS dispatcher excess MVS TCB scan. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates then an excess TCB scan has not happened yet. Reset characteristic: not reset
Number of Excess TCB Scans	DSGXSCNS	is the number of CICS dispatcher excess MVS TCB scans. Reset characteristic: reset to zero
Last Excess TCB Scan–No TCB Detached	DSGLXSND	The date and time of the last CICS dispatcher excess MVS TCB scan that did not detach any TCBs. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates then an excess TCB scan has not happened yet. Reset characteristic: not reset
Number of Excess TCB Scans–No TCB Detached	DSGXSCNN	is the number of excess MVS TCB scans that resulted in no MVS TCBs being detached by the CICS dispatcher. Reset characteristic: reset to zero
Number of Excess TCBs Detached	DSGXTCBD	is the total number of MVS TCBs that have been detached by the CICS dispatcher's excess MVS TCB management processing. Reset characteristic: reset to zero

Table 58. Dispatcher domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Average Excess TCBs Detached per Scan	Not Applicable	is the average number of MVS TCBs that have been detached by each scan of the CICS dispatcher's excess MVS TCB management processing. Reset characteristic: reset to zero
Number of CICS TCB MODEs	DSGASIZE	is the current number of CICS TCB modes in which the CICS dispatcher is managing MVS task control blocks (TCBs) in the system. Reset characteristic: not reset
Number of CICS TCB POOLs	DSGPSIZE	is the number of TCB pools in which the CICS dispatcher is managing MVS task control blocks (TCBs) in the system under which the CICS dispatcher runs. Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

Dispatcher domain: TCB Mode statistics

You can retrieve dispatcher domain TCB mode statistics by using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** system command. They are mapped by the DFHDSGDS DSECT.

Two passes are made at the data, producing two TCB Mode statistics tables, because the statistics cannot all be fitted into a single table in the format of the report.

- The first table contains mainly the TCB event information, such as attaches, detaches, and steals, for each mode.
- The second table shows timing information, such as operating system wait time, waits, TCB dispatch, and CPU times. Some fields show accumulated time for all the TCBs in a specific mode. When there can be multiple TCBs, for example L8 open TCBs, be aware that the value of such fields is a total that might exceed the time since the statistics were last reset.

The following fields are mapped by the DSGTCBM DSECT in the DFHDSGDS DSECT. The DSGTCBM DSECT is repeated for each mode of TCB in CICS (DSGASIZE). For a list of modes of TCB, see "Interpreting dispatcher statistics" on page 83.

Table 59. Dispatcher domain: TCB Mode statistics - Pass 1

DFHSTUP name	Field name	Description
TCB Mode	DSGTCBNM	The name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, orT8. Reset characteristic: not reset

Table 59. Dispatcher domain: TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Field name	Description
Open	DSGTCBMD	Indicates whether the CICS dispatcher TCB mode is open, not open, or unknown. A TCB mode of type 'unknown' indicates that this TCB mode has not been activated. Reset characteristic: not reset
TCB Pool	DSGTCBMP	The name of the TCB pool in which this TCB mode is defined, either N/A, OPEN, SSL, THRD, or XP. Reset characteristic: not reset
TCBs Attached – Current	DSGTCBCA	The current number of MVS TCBs attached in this TCB mode. Reset characteristic: not reset
TCBs Attached – Peak	DSGTCBPA	The peak number of MVS TCBs attached in this TCB mode. Reset characteristic: reset to current value
TCBs In Use – Current	DSGTCBCU	The current number of MVS TCBs in use in this TCB mode. Reset characteristic: not reset
TCBs In Use – Peak	DSGTCBPU	The peak number of MVS TCBs in use in this TCB mode. Reset characteristic: reset to current value
Dispatchable Queue - Current	DSGTMCDQ	The current number of dispatchable tasks queued for the TCB. Reset characteristic: not reset
Dispatchable Queue - Peak	DSGTMPDQ	The peak number of dispatchable tasks that have been queued for the TCB. Reset characteristic: reset to current
Dispatchable Queue - Average	DSGTMADQ	The average number of dispatchable tasks that have been queued for the TCB. Reset characteristic: reset to current
TCB Attaches	DSGNTCBA	The number of MVS TCBs that have been attached in this TCB mode. Reset characteristic: reset to zero

Table 59. Dispatcher domain: TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Field name	Description
Detached Unclean	DSGTCBDU	The number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because the CICS transaction that was associated with the TCB has abended. Reset characteristic: reset to zero
Detached Stolen	DSGTCBDS	The number of MVS TCBs that have been, or are in the process of being, stolen from this TCB mode because they are required by another TCB mode. Reset characteristic: reset to zero
Detached Excess	DSGTCBDX	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode because of the dispatcher excess TCB management processing. Reset characteristic: reset to zero
Detached Other	DSGTCBDO	The number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode. They are detached because, for example, the limit for the number of TCBs allowed in the TCB pool has been lowered, or too many TCBs are attached in relation to the number of TCBs in use. Reset characteristic: reset to zero
TCB Steals	DSGTCBST	The number of MVS TCBs that have been stolen from other TCB modes. Reset characteristic: reset to zero
TCB Mismatches	DSGTCBMM	The number of MVS TCB mismatches that have occurred for this TCB mode. Reset characteristic: reset to zero

Table 60. Dispatcher domain: TCB Mode statistics - Pass 2

DFHSTUP name	Field name	Description
Mode	DSGTGBM	The name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, orT8. Reset characteristic: not reset

Table 60. Dispatcher domain: TCB Mode statistics - Pass 2 (continued)

DFHSTUP name	Field name	Description
TCBs Attached – Current	DSGTCBCA	The current number of MVS TCBs attached in this TCB mode. Reset characteristic: not reset
TCBs Attached – Peak	DSGTCBPA	The peak number of MVS TCBs attached in this TCB mode. Reset characteristic: not reset
TCB Attaches	DSGNTCBA	The number of MVS TCBs that have been attached in this TCB mode. Reset characteristic: reset to zero
Attach Failures	DSGTCBAF	The number of MVS TCB attach failures that have occurred in this TCB mode. Reset characteristic: reset to zero
MVS Waits	DSGSYSW	The number of MVS waits that occurred on TCBs in this mode. Reset characteristic: reset to zero
Accum Time in MVS wait	DSGTWT	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 *days-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	The accumulated real time that TCBs in this mode have been dispatched by MVS; that is, the total time used between the end of an MVS wait issued by the dispatcher and the start of the subsequent wait issued by the dispatcher. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

Table 60. Dispatcher domain: TCB Mode statistics - Pass 2 (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	DSGTCT	The accumulated CPU time taken for the DS task, that is, the processor time used by TCBs in this mode while running 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	The accumulated CPU time taken for all the TCBs that are, or have been, attached in this TCB mode; that is, the total time that TCBs in this mode have been running. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*; however, the DSECT field contains the time as a store clock (STCK) value. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Dispatcher domain: TCB Pool statistics

You can access dispatcher domain TCB pool statistics by using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** system command. They are mapped by the DFHDSGDS DSECT.

Statistics are produced for each TCB pool: the OPENAPI TCB pool, the SSL TCB pool, the JVM server THRD TCB pool, and the XP TCB pool.

The following fields are mapped by the DSGTCBP DSECT in the DFHDSGDS DSECT. The DSGTCBP DSECT is repeated for each TCB pool in CICS (DSGPSIZE).

Table 61. Dispatcher domain: TCB Pool statistics

DFHSTUP name	Field name	Description
TCB Pool	DSGTCBPN	The name of the CICS TCB pool, either OPEN, SSL, THRD, or XP. Reset characteristic: not reset
Current TCBs attached in this TCB Pool	DSGCNUAT	The current number of TCBs attached in the TCB modes that are in this TCB pool. Reset characteristic: not reset
Peak TCBs attached in this TCB Pool	DSGPNUAT	The peak number of TCBs attached in the TCB modes that are in this TCB pool. Reset characteristic: reset to current

Table 61. Dispatcher domain: TCB Pool statistics (continued)

DFHSTUP name	Field name	Description
Current TCBs in use in this TCB Pool	DSGCNUUS	The current number of CICS TCBs attached in this TCB pool and being used. Reset characteristic: not reset
Peak TCBs in use in this TCB Pool	DSGPNUUS	The peak number of CICS TCBs used that were attached in this TCB pool. Reset characteristic: reset to current value
Max TCB Pool limit	DSGMXTCB	The value for the maximum number of TCBs allowed in this pool: • The **MAXOPENTCBS** system initialization parameter, if specified, sets the value for the open TCB pool. If the MAXOPENTCBS system initialization is not specified, CICS sets the limit for the L8 and L9 mode open TCB pool automatically based on the maximum number of tasks specified for the CICS region (the MXT value), using the following formula: `(2 * MXT Value) + 32`. • The **MAXSSLTCBS** system initialization parameter specifies the value for the SSL TCB pool. • The **MAXTHRDTCBS** system initialization parameter specifies the value for the JVM server THRD TCB pool. The number of threads reserved for each JVM server is the THREADLIMIT value on the JVMSERVER resource, plus 1, up to a limit of 2000. • The **MAXXPTCBS** system initialization parameter, if specified, sets the value for the XP TCB pool. If the **MAXXPTCBS** system initialization is not specified, CICS sets the limit for the X8 and X9 mode XP TCB pool automatically to a value equal to the maximum number of tasks specified for the CICS region (the MXT value) You can change the maximum value by overriding the appropriate system initialization parameter or by using the SET DISPATCHER command. To change the maximum value of the JVM server, use the SET JVMSERVER command. Reset characteristic: not reset
Time Max TCB Pool Limit last reached	DSGLTCBL	The time at which the pool reached the maximum TCB limit. If the DFHSTUP report shows the time as --:--:--:---- then that indicates that the pool limit has not been reached since the statistics were last reset. Reset characteristic: reset to zero

Table 61. Dispatcher domain: TCB Pool statistics (continued)

DFHSTUP name	Field name	Description
Times at Max TCB Pool Limit	DSGNTCBL	The number of times the system reached the limit for the number of TCBs allowed in this pool: • OPEN TCB pool • SSL TCB pool • THRD TCB pool • XP TCB pool Reset characteristic: reset to zero
Total Requests delayed by Max TCB Pool Limit	DSGTOTNW	The total number of TCB requests delayed because the system reached the limit for the number of TCBs allowed in this pool. Reset characteristic: reset to zero
Total Max TCB Pool Limit delay time	DSGTOTWL	The total time that TCB requests were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Reset characteristic: reset to zero
Current Requests delayed by Max TCB Pool Limit	DSGCURNW	The number of TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Reset characteristic: not reset
Current Max TCB Pool Limit delay time	DSGCURWT	The current delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Reset characteristic: not reset
Peak Requests delayed by Max TCB Pool Limit	DSGPEANW	The peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Reset characteristic: not reset
Total Number of TCB Mismatch waits	DSGMMWTS	The total number of TCB mismatch waits; that is, TCB requests that waited because no TCB was available that matched the request, but at least one non-matching TCB was free. Reset characteristic: reset to zero
Total TCB Mismatch wait time	DSGMMWTM	The total time spent in TCB mismatch waits by TCB requests using this pool. Reset characteristic: reset to zero

Table 61. Dispatcher domain: TCB Pool statistics (continued)

DFHSTUP name	Field name	Description
Current TCB Mismatch waits	DSGCMMWS	The current number of TCB mismatch waits by TCB requests using this pool. Reset characteristic: not reset
Current TCB Mismatch wait time	DSGCMMWT	The current wait time for current TCB mismatch waits by TCB requests using this pool. Reset characteristic: not reset
Peak TCB mismatch waits	DSGPMMWS	The peak number of TCB mismatch waits by TCB requests using this pool. Reset characteristic: reset to current value
Requests delayed by MVS storage constraint	DSGTOTMW	The total number of MVS storage requests that have waited because no TCB was available, and none was created because of MVS storage constraints. Reset characteristic: reset to zero
Total MVS storage constraint delay time	DSGTOTMT	The total time spent in MVS storage waits by TCB requests using this pool. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Dispatcher domain: MVS TCB statistics

You can retrieve dispatcher domain MVS TCB statistics by using the **EXEC CICS EXTRACT STATISTICS DISPATCHER**, **EXTRACT STATISTICS MVSTCB**, and **INQUIRE MVSTCB** system commands. The statistics data is mapped by the DFHDSGDS, DFHDSTDS, and DFHDSRDS DSECTs.

Reset characteristics: These statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high-water and low-water marks (max, min and highest, lowest) are reset to current; counts are reset to zero.

Table 62. Dispatcher domain: MVS TCB statistics

DFHSTUP Name	Field Name	Description
Dispatcher MVS TCB		
Dispatcher Start Time and Date	DSGLSTRT	The local time and date at which the CICS dispatcher started.
Address Space Accumulated CPU Time	MVS field ASCBEJST	The accumulated CPU time since reset for this CICS address space. If the time is greater than 24 hours, this time is prefixed with the number of days.

Table 62. Dispatcher domain: MVS TCB statistics (continued)

DFHSTUP Name	Field Name	Description
Address Space Accumulated SRB Time	MVS field ASCBSRBT	The accumulated SRB time since reset for this CICS address space.
Address Space CPU Time (Since Reset)	DSGEJST	The accumulated CPU time for this CICS address space.
Address Space SRB Time (Since Reset)	DSGSRBT	The accumulated SRB time for this CICS address space.
Current number of CICS TCBs	DSTDS_CICSTCB_COUNT	The current number of CICS TCBs in the address space.
Current CICS TCB CPU time	DSTDS_CICSTCB_CPUTIME	The total CPU time so far for the currently attached CICS TCBs.
Current CICS TCB Private Stg below 16MB	DSTDS_CICSTCB_STG_BELOW	The total private storage below 16 MB allocated to CICS TCBs.
Current CICS TCB Private Stg below 16MB in use	DSTDS_CICSTCB_STG_BELOW_INUSE	The total private storage below 16 MB in use by CICS TCBs. [1]
Current CICS TCB Private Stg above 16MB	DSTDS_CICSTCB_STG_ABOVE	The total private storage above 16 MB allocated to CICS TCBs.
Current CICS TCB Private Stg above 16MB in use	DSTDS_CICSTCB_STG_ABOVE_INUSE	The total private storage above 16 MB in use by CICS TCBs. [1]
Current number of non-CICS TCBs	DSTDS_NONCICSTCB_COUNT	The current number of non-CICS TCBs in the address space.
Current non-CICS TCB CPU time	DSTDS_NONCICSTCB_CPUTIME	The total CPU time so far for the currently attached non-CICS TCBs.
Current non-CICS TCB Private Stg below 16MB	DSTDS_NONCICSTCB_STG_BELOW	The total private storage below 16 MB allocated to non-CICS TCBs.
Current non-CICS TCB Private Stg below 16MB in use	DSTDS_NONCICSTCB_STG_BELOW_INUSE	The total private storage below 16 MB in use by non-CICS TCBs.
Current non-CICS TCB Private Stg above 16MB	DSTDS_NONCICSTCB_STG_ABOVE	The total private storage above 16 MB allocated to non-CICS TCBs.
Current non-CICS TCB Private Stg above 16MB in use	DSTDS_NONCICSTCB_STG_ABOVE_INUSE	The total private storage above 16 MB in use by non-CICS TCBs.
TCB Address	DSRDS_TCB_ADDRESS	The address of the MVS TCB.
TCB Name	DSRDS_TCB_NAME	The name of the MVS TCB (if known to CICS).
CICS TCB	DSRDS_TCB_TYPE	The type of TCB, CICS or non-CICS.
Current TCB CPU Time	DSRDS_TCB_CPUTIME	The total CPU time so far for this TCB.
Current TCB Private Stg Below 16MB Allocated	DSRDS_TCB_STG_BELOW	The total private storage below 16 MB allocated to this TCB.
Current TCB Private Stg Below 16MB In Use	DSRDS_TCB_STG_BELOW_INUSE	The total private storage below 16 MB in use by this TCB.
Current TCB Private Stg Above 16MB Allocated	DSRDS_TCB_STG_ABOVE	The total private storage above 16 MB allocated to this TCB.

Table 62. Dispatcher domain: MVS TCB statistics (continued)

DFHSTUP Name	Field Name	Description
Current TCB Private Stg Above 16MB In Use	DSRDS_TCB_STG_ABOVE_INUSE	The total private storage above 16 MB in use by this TCB.
Task Number	DSRDS_TCB_CICS_TASK	The CICS task number currently associated with this TCB. None means there are no CICS transactions currently assigned to this TCB.
Tran ID	EXEC CICS INQUIRE TASK() TRANSACTION()	Transaction ID of the task currently associated with this TCB, if any.
Task Status	EXEC CICS INQUIRE TASK() RUNSTATUS()	Status of the task currently associated with this TCB, if any.
Mother TCB	DSRDS_TCB_MOTHER	Address of mother TCB.
Sister TCB	DSRDS_TCB_SISTER	Address of sister TCB.
Daughter TCB	DSRDS_TCB_DAUGHTER	Address of daughter TCB.

Note:

1. The statistics for storage in use show the amount of storage that tasks obtain by using a GETMAIN request. This might be less than the amount of storage allocated to the TCBs, because storage is always allocated to TCBs in page multiples (4096 bytes).

Related information:

EXTRACT STATISTICS in Reference > System programming

INQUIRE MVSTCB in Reference > System programming

Dispatcher domain: Summary global statistics

Dispatcher domain Summary statistics are not available online.

Table 63. Dispatcher domain: Summary global statistics

DFHSTUP name	Description
Dispatcher Start Date and Time	is the date and time at which the CICS dispatcher started. This value can be used as an approximate date and time at which CICS started. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds.decimals* at the local time; however, the DSECT field contains the time as a local store clock (STCK) value.
Address Space CPU Time	is the total CPU time taken by the CICS address space. The DFHSTUP report expresses this as *days-hours:minutes:seconds.decimals*
Address Space SRB Time	is the total SRB time taken by the CICS address space. The DFHSTUP report expresses this as *days-hours:minutes:seconds.decimals*
Peak number of dispatcher tasks	is the peak number of dispatcher 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.
Peak ICVR time (msec)	is the peak ICVR time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically.

Table 63. Dispatcher domain: Summary global statistics (continued)

DFHSTUP name	Description
Peak ICVTSD time (msec)	is the peak ICVTSD time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically.
Peak PRTYAGE time (msec)	is the peak PRTYAGE time value (expressed in *milliseconds*) specified in the SIT, or as an override, or changed dynamically.
Peak MRO (QR) Batching (MROBTCH) value	is the peak MROBTCH value specified in the SIT, or as an override, or changed dynamically.
Number of Excess TCB scans	is the total number of CICS dispatcher excess MVS TCB scans.
Excess TCB scans – No TCB detached	is the total number of CICS dispatcher excess MVS TCB scans which resulted in no MVS TCB being detached.
Number of Excess TCBs detached	is the total number of MVS TCBs that have been detached by the CICS dispatcher's excess MVS TCB management processing.
Average Excess TCBs Detached per Scan	is the average number of MVS TCBs that have been detached by each scan of the CICS dispatcher's excess MVS TCB management processing.
Number of CICS TCB MODEs	is the number of CICS dispatcher TCB modes.
Number of CICS TCB POOLs	is the number of CICS dispatcher TCB pools.

Dispatcher domain: Summary TCB Mode statistics

Dispatcher domain Summary TCB Mode statistics are not available online.

Two passes are made at the data, producing two summary TCB Mode statistics tables, because the statistics cannot all be fitted into a single table in the format of the report. The first table mainly contains the TCB event information, such as attaches, detaches, and steals, for each mode. The second table has timing information, such as operating system wait time, waits, TCB dispatch, and CPU times.

For a list of modes of TCB, see “Interpreting dispatcher statistics” on page 83.

Table 64. Dispatcher domain: Summary TCB Mode statistics - Pass 1

DFHSTUP name	Description
Mode	The name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, orT8.

Table 64. Dispatcher domain: Summary TCB Mode statistics - Pass 1 (continued)

DFHSTUP name	Description
Open	Indicates whether the CICS dispatcher TCB mode is open, not open, or unknown. A TCB mode of type `Unk` indicates that this TCB mode has not been activated.
TCB Pool	The name of the CICS TCB pool, either N/A, OPEN, THRD, SSL, or XP.
Peak TCBs Attached	The peak number of MVS TCBs attached in this TCB mode.
Peak TCBs In Use	The peak number of MVS TCBs attached and in use in this TCB mode.
TCB Attaches	The number of MVS TCBs that have been attached in this TCB mode.
Detached Unclean	The total number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because the CICS transaction that was associated with the TCB has abended.
Detached Stolen	The total number of MVS TCBs that have been stolen, or are in the process of being stolen, from this TCB mode because they are required by another TCB mode.
Detached Excess	The total number of MVS TCBs that have been, or are in the process of being, detached from this TCB mode because of the dispatcher excess TCB management processing.
Detached Other	The total number of MVS TCBs that have been detached, or are in the process of being detached, from this TCB mode. They are being detached, for example, the limit for the number of TCBs allowed in the TCB pool has been lowered, or too many TCBs are attached in relation to the number of TCBs in use.
TCB Steals	The total number of MVS TCBs that have been stolen from other TCB modes.
TCB Mismatches	The total number of MVS TCB mismatches that have occurred for this TCB mode.

Table 65. Dispatcher domain: Summary TCB Mode statistics - Pass 2

DFHSTUP name	Description
Mode	The name of the CICS dispatcher TCB mode, either QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, orT8.
Peak TCBs Attached	The peak number of MVS TCBs attached in this TCB mode.
Peak TCBs In Use	The peak number of MVS TCBs attached and in use in this TCB mode.

Table 65. Dispatcher domain: Summary TCB Mode statistics - Pass 2 (continued)

DFHSTUP name	Description
TCB Attaches	The number of MVS TCBs that have been attached in this TCB mode.
Attach Failures	The total number of MVS TCB attach failures that have occurred in this TCB mode.
MVS Waits	The total number of MVS waits that occurred on this TCB mode.
Total Time in MVS wait	The total real time that the TCBs in this mode were in an MVS wait. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Total Time Dispatched	The total real time that the TCBs in this mode were dispatched by MVS. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.
Total CPU Time / TCB	The total CPU time taken for all the TCBs in this mode. The DFHSTUP report expresses this time as *days-hours:minutes:seconds.decimals*.

Dispatcher domain: Summary TCB Pool statistics

Statistics are produced for each TCB pool: the OPENAPI TCB pool, the SSL TCB pool, the JVM server THRD TCB pool, and the XP TCB pool.

Table 66. Dispatcher domain: Summary TCB Pool statistics

DFHSTUP name	Description
TCB Pool	The name of the CICS TCB pool, either OPEN, SSL, THRD, or XP.
Peak TCBs attached in this TCB Pool	The peak number of TCBs attached in the TCB modes that are in this TCB pool.
Peak TCBs in use in this TCB Pool	The peak number of CICS TCBs used that were attached in this TCB pool.

Table 66. Dispatcher domain: Summary TCB Pool statistics (continued)

DFHSTUP name	Description
Max TCB Pool limit	The value for the maximum number of TCBs allowed in this pool: • The **MAXOPENTCBS** system initialization parameter, if specified, sets the value for the open TCB pool. If the MAXOPENTCBS system initialization is not specified, CICS sets the limit for the L8 and L9 mode open TCB pool automatically based on the maximum number of tasks specified for the CICS region (the MXT value), using the following formula: `(2 * MXT Value) + 32`. • The **MAXSSLTCBS** system initialization parameter specifies the value for the SSL TCB pool. • The **MAXTHRDTCBS** system initialization parameter specifies the value for the JVM server THRD TCB pool. The number of threads reserved for each JVM server is the THREADLIMIT value on the JVMSERVER resource, plus 1, up to a limit of 2000. • The **MAXXPTCBS** system initialization parameter, if specified, sets the value for the XP TCB pool. If the **MAXXPTCBS** system initialization is not specified, CICS sets the limit for the X8 and X9 mode XP TCB pool automatically to a value equal to the maximum number of tasks specified for the CICS region (the MXT value)
Times at Max TCB Pool Limit	The total number of times that the limit for the number of TCBs allowed in this pool has been reached: • OPEN TCB pool • SSL TCB pool • THRD TCB pool • XP TCB pool
Total Requests delayed by Max TCB Pool Limit	The total number of TCB requests that have been delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Total Max TCB Pool Limit delay time	The total time spent waiting by those tasks that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Average Max TCB Pool Limit delay time	The average time spent waiting by those tasks that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Peak Requests delayed by Max TCB Pool Limit	The peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool.
Total number of TCB Mismatch waits	The total number of TCB mismatch waits; that is, TCB requests that waited because no TCB matching the request was available, but at least one non-matching TCB was free.
Total TCB Mismatch wait time	The total time spent in TCB mismatch waits by TCB requests using this pool.
Average TCB Mismatch wait time	The average time spent in TCB mismatch waits by TCB requests using this pool.
Peak TCB Mismatch waits	The peak number of TCB mismatch waits by TCB requests using this pool.

Table 66. Dispatcher domain: Summary TCB Pool statistics (continued)

DFHSTUP name	Description
Requests delayed by MVS storage constraint	The total number of MVS storage requests that have waited because no TCB was available, and none could be created because of MVS storage constraints.
Total MVS storage constraint delay time	The total time spent in MVS storage waits by TCB requests using this pool.

Document template statistics

Document templates are used in CICS web support to produce the body of HTTP messages. They can be specified in a URIMAP definition to provide a static response to a web client request, or they can be used by an application program to make an HTTP request or response, or for other uses.

Usage statistics are provided for each document template. A DFH0STAT report lists each document template that is defined in the CICS region, and gives information about its source and usage.

For more information about the document template statistics report, see "Document Templates report" on page 825.

Related reference:

"Document Templates report" on page 825
The Document Templates report is produced using the **EXEC CICS EXTRACT STATISTICS DOCTEMPLATE** command and the **EXEC CICS INQUIRE DOCTEMPLATE** command. The statistics data is mapped by the DFHDHDDS DSECT.

Document templates: Resource statistics

You can access document templates resource statistics by using the **EXEC CICS EXTRACT STATISTICS DOCTEMPLATE** system command. They are mapped by the DFHDHDDS DSECT.

The resource information gives details of various attribute settings of each DOCTEMPLATE resource, and the usage of the document template.

Table 67. Document templates: Resource statistics

DFHSTUP name	Field name	Description
DOCTEMPLATE name	DHD_DOCTEMPLATE_NAME	The name of the DOCTEMPLATE resource definition. Reset characteristic: not reset

Table 67. Document templates: Resource statistics (continued)

DFHSTUP name	Field name	Description
Template name	DHD_TEMPLATE_NAME	The name by which the template is known to application programs (the TEMPLATENAME attribute in the DOCTEMPLATE resource definition). Reset characteristic: not reset
Append crlf	DHD_APPEND_CRLF	Whether CICS appends carriage-return line-feed to each logical record of the template. Reset characteristic: not reset
Template contents	DHD_TEMPLATE_CONTENTS	The format of the contents of the template, either binary or EBCDIC. Reset characteristic: not reset
Template type	DHD_TEMPLATE_TYPE	The type for the source of the document template, which can be an exit program, a CICS file name for a data set, an zFS file, a member of a PDS, a program, a transient data queue, or a temporary storage queue. Reset characteristic: not reset
Template type name	DHD_TEMPLATE_EXIT_PROGRAM DHD_TEMPLATE_FILE_NAME DHD_TEMPLATE_PROGRAM_NAME DHD_TEMPLATE_PDS_MEMBER DHD_TEMPLATE_TDQUEUE DHD_TEMPLATE_TSQUEUE DHD_TEMPLATE_HFSFILE	The name for the source of the document template, such as a program name or zFS file name. Reset characteristic: not reset

Table 67. Document templates: Resource statistics (continued)

DFHSTUP name	Field name	Description
Template cache size	DHD_TEMPLATE_CACHE_SIZE	The amount of storage required for a cached copy of the document template. • Before the first use of the template, this field is zero. • This field is always zero for templates in a CICS program, which are never cached, and for templates in an exit program if they are not specified for caching. Reset characteristic: not reset
Use count	DHD_TEMPLATE_USE_COUNT	The total number of times the document template was referenced for any reason. Reset characteristic: reset to zero
Newcopy count	DHD_TEMPLATE_NEWCOPIES	The number of times the SET DOCTEMPLATE NEWCOPY command was issued for this document template. Reset characteristic: reset to zero
Read count	DHD_TEMPLATE_READ_COUNT	The number of times the document template was read from the source. This read happens on the first use, including the first reference after deletion from the cache, or by a SET DOCTEMPLATE NEWCOPY command. Reset characteristic: reset to zero

Table 67. Document templates: Resource statistics (continued)

DFHSTUP name	Field name	Description
Cache copy used	DHD_TEMPLATE_CACHE_USED	The number of times an application used the cached copy of the document template. Reset characteristic: reset to zero
Cache copy deleted	DHD_TEMPLATE_CACHE_DELETED	The number of times the cached copy of the document template was deleted because of a short-on-storage condition. Reset characteristic: reset to zero
Not in DFHSTUP report	DHD_TEMPLATE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	DHD_TEMPLATE_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	DHD_TEMPLATE_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	DHD_TEMPLATE_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset

Table 67. Document templates: Resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	DHD_TEMPLATE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	DHD_TEMPLATE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	DHD_TEMPLATE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Document templates: Summary resource statistics

Document templates summary resource statistics are not available online.

The resource information gives details of various attribute settings of each DOCTEMPLATE resource definition, and the usage of the document template.

Table 68. Document templates: Summary resource statistics

DFHSTUP name	Description
DOCTEMPLATE name	The name of the DOCTEMPLATE resource definition.
Template name	The name by which the template is known to application programs (the TEMPLATENAME attribute in the DOCTEMPLATE resource definition).
Append crlf	Whether CICS appends carriage-return line-feed to each logical record of the template.

Table 68. Document templates: Summary resource statistics (continued)

DFHSTUP name	Description
Template contents	The format of the contents of the template, either binary or EBCDIC.
Template type	The name of the DOCTEMPLATE resource definition.
[Template type] name	The name for the source of the document template, such as a program name or z/OS UNIX file name.
Template cache size	The amount of storage required for a cached copy of the document template. In the summary resource statistics, this value shows the most recent non-zero template size.
Use count	The total number of times the document template was referenced for any reason.
Newcopy count	The number of times the SET DOCTEMPLATE NEWCOPY command was issued for this document template.
Read count	The number of times the document template was read from the source.
Cache copy used	The number of times an application used the cached copy of the document template.
Cache copy deleted	The number of times the cached copy of the document template was deleted because of a short-on-storage condition.

Dump domain statistics

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

Dump domain: System dump statistics

The dump domain collects global and resource statistics for both system and transaction dumps which occur during the CICS run.

Related concepts:

"Dump domain statistics"
Both transaction and system dumps are very expensive and should be thoroughly investigated and eliminated.

Dump domain: Global statistics - system dump

You can retrieve system dump global statistics by using the `EXEC CICS EXTRACT STATISTICS SYSDUMPCODE` system command. They are mapped by the DFHSDGDS DSECT.

These statistics fields contain the global data collected by the dump domain for system dumps.

Table 69. Dump domain: Global statistics - system dump

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

Related information:

EXTRACT STATISTICS in Reference > System programming

Dump domain: Resource statistics - system dump

You can retrieve system dump resource statistics by using the **EXEC CICS EXTRACT STATISTICS SYSDUMPCODE** system command. They are mapped by the DFHSDRDS DSECT.

These statistics fields contain the data collected by the dump domain for system dumps, by dump code

Table 70. Dump domain: Resource statistics - system dump

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 *CICS Messages and Codes*. 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

Table 70. Dump domain: Resource statistics - system dump (continued)

DFHSTUP name	Field name	Description
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

Related information:

EXTRACT STATISTICS in Reference > System programming

Dump domain: Summary global statistics - system dump

Dump domain Summary global statistics are not available online.

Table 71. Dump domain: Summary system dump global statistics

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 resource statistics - system dump

Dump domain Summary resource statistics are not available online.

Table 72. Dump domain: Summary resource statistics - system dump

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 *CICS Messages and Codes*.

Table 72. Dump domain: Summary resource statistics - system dump (continued)

DFHSTUP name	Description
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.

Dump domain: Transaction dump statistics

The dump domain collects global and resource statistics for both system and transaction dumps which occur during the CICS run.

Related concepts:

"Dump domain statistics" on page 490
Both transaction and system dumps are very expensive and should be thoroughly investigated and eliminated.

Dump domain: Global statistics - transaction dump

You can retrieve transaction dump global statistics by using the **EXEC CICS EXTRACT STATISTICS TRANDUMPCODE** system command. They are mapped by the DFHTDGDS DSECT.

These statistics fields contain the global data collected by the dump domain for transaction dumps.

Table 73. Dump domain: Global statistics - transaction dump

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

Related information:

EXTRACT STATISTICS in Reference > System programming

Dump domain: Resource statistics - transaction dump

You can retrieve transaction dump resource statistics by using the **EXEC CICS EXTRACT STATISTICS TRANDUMPCODE** system command. They are mapped by the DFHTDRDS DSECT.

These statistics fields contain the data collected by the dump domain for transaction dumps, by dump code

Table 74. Dump domain: Resource statistics - transaction dump

DFHSTUP name	Field name	Description
Dumpcode	TDRCODE	is the transaction dump code. 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
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

Related information:

EXTRACT STATISTICS in Reference > System programming

Dump domain: Summary global statistics - transaction dump

Dump domain Summary global statistics are not available online.

Table 75. Dump domain: Summary global statistics - transaction dump

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 resource statistics - transaction dump

Dump domain Summary resource statistics are not available online.

Table 76. Dump domain: Summary resource statistics - transaction dump

DFHSTUP name	Description
Dumpcode	is the transaction dump code.
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.

Enqueue domain statistics

The enqueue domain collects global statistics for enqueue requests.

Related concepts:

"Interpreting enqueue statistics"
The enqueue domain supports the CICS recovery manager. Enqueue statistics contain the global data collected by the enqueue domain for enqueue requests.

Related reference:

"Enqueue Manager report" on page 827
The Enqueue Manager report is produced using the **EXEC CICS EXTRACT STATISTICS ENQUEUE** command. The statistics data is mapped by the **DFHNQGDS DSECT**.

"Enqueue Models report" on page 828
The Enqueue Models report is produced using the **EXEC CICS INQUIRE ENQMODEL** command.

Interpreting enqueue statistics

The enqueue domain supports the CICS recovery manager. Enqueue statistics contain the global data collected by the enqueue domain for enqueue requests.

Waiting for an enqueue on a resource can add significant delays in the execution of a transaction. The enqueue statistics allow you to assess the impact of waiting for enqueues in the system and the impact of retained enqueues on waiters. Both the current activity and the activity since the last reset are available.

Enqueue domain: Global statistics - enqueue requests

You can retrieve enqueue request statistics by using the **EXEC CICS EXTRACT STATISTICS ENQUEUE** system command. They are mapped by the DFHNQGDS DSECT.

These statistics fields contain the global data collected by the enqueue domain for enqueue requests.

Table 77. Enqueue domain: Global statistics - enqueue requests

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	NQGNPOOL	is the number of enqueue pools. Reset characteristic: not reset
ENQ Poolname	NQGPOOL	is the enqueue pool id. Reset characteristic: not reset
ENQs Issued	NQGTNQSI	is the total number of enqueue requests issued. Reset characteristic: reset to zero
ENQs Waited	NQGTNQSW	is the total number of enqueue requests that had waited due to the enqueues being held. This is a subset of NQGTNQSI. Note that this value does not include the enqueue requests currently waiting (see NQGCNQSW). Reset characteristic: reset to zero
Enqueue Waiting time	NQGTNQWT	is the total waiting time for the enqueue requests that waited (NQGTNQSW). Note that this value does not include the time for the enqueue requests currently waiting (see NQGCNQWT). Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	NQGCNQSW	is the current number of enqueue requests waiting. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGCNQWT	is the total waiting time for the enqueue requests that are currently waiting due to the enqueue being held by another transaction. Reset characteristic: not reset
Sysplex Waited	NQGGNQSW	is the total number of sysplex enqueue requests that had waited due to the enqueues being held. Reset characteristic: reset to zero
Sysplex Waiting time	NQGGNQWT	is the total waiting time for the sysplex enqueue requests that waited (NQGGNQSW). Reset characteristic: reset to zero

Table 77. Enqueue domain: Global statistics - enqueue requests (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	NQGSNQSW	is the current number of sysplex enqueues waiting. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGSNQWT	is the total waiting time for the sysplex enqueues that are currently waiting (NQGSNQSW). Reset characteristic: not reset
Enqueues Retained	NQGTNQSR	is the total number of enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueues that are currently retained (see NQGCNQSR). For more information about shunted UOWs see "Recovery manager statistics" on page 667. Reset characteristic: reset to zero
Enqueue Retention	NQGTNQRT	is the total retention time for the enqueues that were retained due to the owning UOW being shunted. Note that this value does not include the enqueue retention time for those currently retained (see NQGCNQRT). For more information about shunted UOWs see "Recovery manager statistics" on page 667. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	NQGCNQSR	is the current number of enqueues retained. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	NQGCNQRT	is the current enqueue retention time. Reset characteristic: not reset
Immediate-rejection		
–Enqbusy	NQGTIRJB	is the total number of enqueue requests that were immediately rejected due to the enqueue being busy (ENQBUSY response). This value is a subset of the total number of enqueue requests (NQGTNQSI). Reset characteristic: reset to zero

Table 77. Enqueue domain: Global statistics - enqueue requests (continued)

DFHSTUP name	Field name	Description
–Retained	NQGTIRJR	is the total number of enqueue requests that were immediately rejected due to the enqueue being in a retained state. This value is a subset of the total number of enqueue requests (NQGTNQSI). Reset characteristic: reset to zero
Waiting rejection –Retained	NQGTWRJR	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero
–Operator	NQGTWPOP	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero
–Timeout	NQGTWPTO	is the total number of waiting enqueue requests that were rejected due to the timeout value (DTIMEOUT) being exceeded. This value is a subset of the number of enqueue requests that waited (NQGTNQSW). Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Enqueue domain: Summary global statistics

Enqueue domain Summary global statistics are not available online.

These statistics fields contain the enqueue summary global data.

Table 78. Enqueue domain: Summary global statistics

DFHSTUP name	Description
ENQ Poolname	is the enqueue pool id.
ENQs Issued	is the total number of enqueue requests that were issued.
ENQs Waited	is the total number of enqueues requests that waited.
Enqueue Waiting time	is the waiting time for enqueue requests that waited.
Sysplex Waited	is the total number of sysplex enqueue requests that had waited due to the enqueues being held.
Sysplex Waiting time	is the total waiting time for the sysplex enqueue requests that waited.
ENQs Retained	is the total number of enqueues retained.
Enqueue Retention	is the enqueue retention time.
Immediate-rejection	
–Enqbusy	is the total number of enqueue requests that were immediately rejected ENQBUSY.

Table 78. Enqueue domain: Summary global statistics (continued)

DFHSTUP name	Description
–Retained	is the total number of enqueue requests immediately rejected due to the enqueue being in a retained state.
Waiting rejection	
–Retained	is the total number of waiting enqueue requests that were rejected due to the required enqueue moving into a retained state.
–Operator	is the total number of waiting enqueue requests that were rejected due to the operator purging the waiting transaction.
–Timeout	is the total number of waiting enqueue requests that were rejected due to the timeout value being exceeded.

Event processing statistics

Related reference:

"CAPTURESPEC statistics"
Shows information and statistics about the capture specifications for each event.

"EVENTBINDING statistics" on page 503
Shows information and statistics about each event binding.

CAPTURESPEC statistics

Shows information and statistics about the capture specifications for each event.

Related reference:

"EPADAPTER statistics" on page 501
Shows information and statistics about EP adapters.

"EVENTBINDING statistics" on page 503
Shows information and statistics about each event binding.

"EVENTPROCESS statistics" on page 507
Shows information and statistics about event processing.

"CAPTURESPEC report" on page 829
The CAPTURESPEC report shows information and statistics about the capture specifications for each event. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING**, **EXEC CICS INQUIRE CAPTURESPEC**, **EXEC CICS EXTRACT STATISTICS EVENTBINDING**, and **CAPTURESPEC** commands.

CAPTURESPEC: Resource statistics

You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

Table 79. CAPTURESPEC: Resource statistics

DFHSTUP name	Field name	Description
EVENTBINDING Name	ECC_EVENTBINDING_NAME	The name of the associated event binding. Reset characteristic: not reset
CAPTURESPEC Name	ECC_ CAPTURESPEC_NAME	The name of the capture specification. Reset characteristic: not reset
CAPTURESPEC Capture point	ECC_CAPTURE_POINT	The capture point associated with the capture specification. Reset characteristic: not reset

Table 79. CAPTURESPEC: Resource statistics (continued)

DFHSTUP name	Field name	Description
CAPTURESPEC Capture point type	ECC_CAPTURE_POINT_TYPE	The capture point type associated with the capture specification. Reset characteristic: not reset
CAPTURESPEC Event name	ECC_EVENT_NAME	The associated business event name. Reset characteristic: not reset
CAPTURESPEC Events Captured	ECC_EVENTS_CAPTURED	The total number of events captured. Reset characteristic: reset to zero
CAPTURESPEC Capture Failures	ECC_CAPTURE_FAILURES	The number of capture failures, recorded by capture specification. When displayed, this statistic is totaled by event binding. Reset characteristic: reset to zero

Related reference:

“EVENTBINDING: Global statistics” on page 504
Shows information and global statistics about event bindings.

“EVENTBINDING: Summary global statistics” on page 506
Shows information and summary global statistics about event bindings.

“EVENTBINDING: Resource statistics” on page 505
Shows information and resource statistics about event bindings.

“EVENTBINDING: Summary resource statistics” on page 506
Shows information and summary resource statistics about event bindings.

“EVENTPROCESS: Global statistics” on page 507
Shows information and global statistics about event processing.

“EVENTPROCESS: Summary global statistics” on page 510
Shows information and summary global statistics about event processing.

Related information:

EXTRACT STATISTICS in Reference > System programming

CAPTURESPEC: Summary resource statistics

Shows summary information and statistics about the capture specifications for each event.

Summary statistics are not available online.

Table 80. CAPTURESPEC: Summary resource statistics

DFHSTUP name	Description
EVENTBINDING Name	The name of the associated event binding.
CAPTURESPEC Name	The name of the capture specification.
CAPTURESPEC Capture point	The capture point associated with the capture specification.
CAPTURESPEC Capture point type	The capture point type associated with the capture specification.
CAPTURESPEC Event name	The associated business event name.
CAPTURESPEC Events Captured	The total number of events captured.
CAPTURESPEC Capture Failures	The number of capture failures, recorded by capture specification. When displayed, this statistic is totaled by event binding.

Related reference:

"EVENTBINDING: Global statistics" on page 504
Shows information and global statistics about event bindings.

"EVENTBINDING: Summary global statistics" on page 506
Shows information and summary global statistics about event bindings.

"EVENTBINDING: Resource statistics" on page 505
Shows information and resource statistics about event bindings.

"EVENTBINDING: Summary resource statistics" on page 506
Shows information and summary resource statistics about event bindings.

"EVENTPROCESS: Global statistics" on page 507
Shows information and global statistics about event processing.

"EVENTPROCESS: Summary global statistics" on page 510
Shows information and summary global statistics about event processing.

EPADAPTER statistics

Shows information and statistics about EP adapters.

Related reference:

"CAPTURESPEC statistics" on page 499
Shows information and statistics about the capture specifications for each event.

"EVENTBINDING statistics" on page 503
Shows information and statistics about each event binding.

"EVENTPROCESS statistics" on page 507
Shows information and statistics about event processing.

EPADAPTER: Resource statistics

Shows information and resource statistics about EP adapters

You can retrieve EPADAPTER statistics by using the **EXEC CICS EXTRACT STATISTICS EVENTPROCESS RESID()** command. They are mapped by the DFHEPRDS DSECT.

Table 81. EPADAPTER: resource statistics

DFHSTUP name	Field name	Description
EPADAPTER Name	EPR_ADAPTER_NAME	The name of the EP adapter. Reset characteristic: not reset
EPADAPTER Type	EPR_ADAPTER_TYPE	The adapter type. Reset characteristic: not reset
EPADAPTER Emission mode	EPR_EMISSION_MODE	The EP adapter emission mode. This identifies whether the EP adapter is for synchronous or asynchronous events. Reset characteristic: not reset
EPADAPTER Number of put events	EPR_PUT_EVENTS	The number of events passed to EP for emission by this adapter. Reset characteristic: not reset

Table 81. EPADAPTER: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	EPR_ADA_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	EPR_ADA_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTBINDING: Global statistics" on page 504
Shows information and global statistics about event bindings.

"EVENTBINDING: Summary global statistics" on page 506
Shows information and summary global statistics about event bindings.

"EVENTBINDING: Resource statistics" on page 505
Shows information and resource statistics about event bindings.

"EVENTBINDING: Summary resource statistics" on page 506
Shows information and summary resource statistics about event bindings.

Related information:

EXTRACT STATISTICS in Reference > System programming

EPADAPTER: Summary resource statistics

Shows information and summary resource statistics about EP adapters.

Summary statistics are not available online.

Table 82. EPADAPTER: summary resource statistics

DFHSTUP name	Description
EPADAPTER Name	The name of the EP adapter.
EPADAPTER Type	The adapter type.
EPADAPTER Emission mode	The EP adapter emission mode. This identifies whether the EP adapter is for synchronous or asynchronous events.
EPADAPTER Number of put events	The number of events passed to EP for emission by this adapter.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTBINDING: Global statistics" on page 504
Shows information and global statistics about event bindings.

"EVENTBINDING: Summary global statistics" on page 506
Shows information and summary global statistics about event bindings.

"EVENTBINDING: Resource statistics" on page 505
Shows information and resource statistics about event bindings.

"EVENTBINDING: Summary resource statistics" on page 506
Shows information and summary resource statistics about event bindings.

EVENTBINDING statistics

Shows information and statistics about each event binding.

Related reference:

"CAPTURESPEC statistics" on page 499
Shows information and statistics about the capture specifications for each event.

"EPADAPTER statistics" on page 501
Shows information and statistics about EP adapters.

"EVENTPROCESS statistics" on page 507
Shows information and statistics about event processing.

“EVENTBINDING report” on page 831
The EVENTBINDING report shows information and statistics about each event binding and the event binding status. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING** and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

EVENTBINDING: Global statistics

Shows information and global statistics about event bindings.

You can retrieve EVENTBINDING global statistics by using the **EXTRACT STATISTICS EVENTBINDING** system command. They are mapped by the DFHECGDS DSECT.

Table 83. EVENTBINDING: Global statistics

DFHSTUP name	Field name	Description
Total event filter operations	ECG_EB_EVENT_FILTER_OPS	The number of event filtering operations. Reset characteristic: reset to zero
Events with disabled EVENTBINDING	ECG_EB_EVENTS_DISABLED	The number of events that were not captured because of a disabled event binding. Reset characteristic: reset to zero
Total events captured	ECG_EB_EVENTS_CAPTURED	The total number of application and system events captured. Reset characteristic: reset to zero
Total system events captured	ECG_SYS_EVENTS_CAPTURED	The number of system events captured. Reset characteristic: reset to zero
Filter operations failed	ECG_FILTER_OPS_FAILED	The number of filtering operations that did not complete because CICS was unable to determine whether an event should have been captured. Reset characteristic: reset to zero
Capture operations failed	ECG_CAPTURE_OPS_FAILED	The number of capture operations that did not complete because CICS determined that an event was required but failed to capture it. Reset characteristic: reset to zero

Related reference:

“CAPTURESPEC: Resource statistics” on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

“CAPTURESPEC: Summary resource statistics” on page 500
Shows summary information and statistics about the capture specifications for each event.

“EVENTPROCESS: Global statistics” on page 507
Shows information and global statistics about event processing.

“EVENTPROCESS: Summary global statistics” on page 510
Shows information and summary global statistics about event processing.

Related information:

EXTRACT STATISTICS in Reference > System programming

EVENTBINDING: Resource statistics

Shows information and resource statistics about event bindings.

You can retrieve EVENTBINDING resource statistics by using the **EXEC CICS EXTRACT STATISTICS EVENTBINDING RESID()** command, They are mapped by the DFHECRDS DSECT.

Table 84. EVENTBINDING: resource statistics

DFHSTUP name	Field name	Description
EVENTBINDING Name	ECR_EVENTBINDING_NAME	The name of the event binding. Reset characteristic: not reset
EVENTBINDING EPADAPTER name	ECR_EPADAPTER_NAME	The name of the EP adapter. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information , see Summary of the resource signature field values in Product overview Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	ECR_EB_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each

event.

"EVENTPROCESS: Global statistics" on page 507
Shows information and global statistics about event processing.

"EVENTPROCESS: Summary global statistics" on page 510
Shows information and summary global statistics about event processing.

Related information:

EXTRACT STATISTICS in Reference > System programming

EVENTBINDING: Summary global statistics

Shows information and summary global statistics about event bindings.

Summary statistics are not available online.

Table 85. EVENTBINDING: Summary global statistics

DFHSTUP name	Description
Total Event Filter operations	The number of event filtering operations.
Events with disabled EVENTBINDING	The number of events that were not captured because of a disabled event binding.
Total Events Captured	The total number of application and system events captured.
Total system events captured	The number of system events captured.
Filter operations failed	The number of filtering operations that did not complete because CICS was unable to determine whether an event should have been captured.
Capture operations failed	The number of capture operations that did not complete because CICS determined that an event was required but failed to capture it.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTPROCESS: Global statistics" on page 507
Shows information and global statistics about event processing.

"EVENTPROCESS: Summary global statistics" on page 510
Shows information and summary global statistics about event processing.

EVENTBINDING: Summary resource statistics

Shows information and summary resource statistics about event bindings.

Summary statistics are not available online.

Table 86. EVENTBINDING: Summary resource statistics

DFHSTUP name	Description
EVENTBINDING Name	The name of the event binding.
EVENTBINDING EPADAPTER Name	The name of the EP adapter.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTPROCESS: Global statistics"
Shows information and global statistics about event processing.

"EVENTPROCESS: Summary global statistics" on page 510
Shows information and summary global statistics about event processing.

EVENTPROCESS statistics

Shows information and statistics about event processing.

Related reference:

"CAPTURESPEC statistics" on page 499
Shows information and statistics about the capture specifications for each event.

"EPADAPTER statistics" on page 501
Shows information and statistics about EP adapters.

"EVENTBINDING statistics" on page 503
Shows information and statistics about each event binding.

EVENTPROCESS: Global statistics

Shows information and global statistics about event processing.

You can retrieve EVENTPROCESS statistics by using the **EXTRACT STATISTICS EVENTPROCESS** system command. They are mapped by the DFHEPGDS DSECT.

Table 87. EVENTPROCESS: global statistics

DFHSTUP name	Field name	Description
Number of put events	EPG_PUT_EVENTS	The number of events passed to the EP component for emission. Reset characteristic: reset to zero
Number of commit forward events	EPG_COMMIT_FORWARD_EVENTS	The number of units of work that have been committed, and that included one or more asynchronous transactional events. Reset characteristic: reset to zero
Number of commit backward events	EPG_COMMIT_BACKWARD_EVENTS	The number of units of work that have been backed out, and that included one or more asynchronous transactional events. Reset characteristic: reset to zero
Current event capture queue	EPG_CURRENT_EVC_QUEUE	The current number of events on the event capture queue. Reset characteristic: not reset
Peak event capture queue	EPG_PEAK_EVC_QUEUE	The peak number of events on the event capture queue. Reset characteristic: reset to current

Table 87. EVENTPROCESS: global statistics (continued)

DFHSTUP name	Field name	Description
Current transactional queue	EPG_CURRENT_TRANS_QUEUE	The current number of events on the transactional queue. Reset characteristic: not reset
Peak transactional queue	EPG_PEAK_TRANS_QUEUE	The peak number of events on the transactional queue. Reset characteristic: reset to current
Number of async normal events	EPG_ASYNC_NORMAL_EVENTS	The number of asynchronous normal priority events. Reset characteristic: reset to zero
Number of async priority events	EPG_ASYNC_PRIORITY_EVENTS	The number of asynchronous high priority events. Reset characteristic: reset to zero
Number of transactional events	EPG_TRANS_EVENTS	The number of transactional events. Reset characteristic: reset to zero
Transaction events discarded	EPG_TRANS_EVENTS_DISCARDED	The number of transactional events discarded. Reset characteristic: reset to zero
Number of synchronous events	EPG_SYNC_EVENTS	The number of synchronous emission events captured. Reset characteristic: reset to zero
Number of sync events failed	EPG_SYNC_EVENTS_FAILED	The number of synchronous emission events that were not emitted. Reset characteristic: reset to zero
Number of dispatcher attaches	EPG_DISPATCHERS_ATTACHED	The number of dispatcher tasks attached. Reset characteristic: reset to zero
Current dispatcher tasks	EPG_CURRENT DISPATCHERS	The current number of dispatcher tasks. Reset characteristic: not reset
Peak dispatcher tasks	EPG_PEAK_DISPATCHERS	The peak number of dispatcher tasks. Reset characteristic: reset to current
Events to WebSphere MQ EP adapter	EPG_WMQ_ADAPTER_EVENTS	The number of events dispatched to the WebSphere MQ EP adapter. Reset characteristic: reset to zero
Events to Transaction EP adapter	EPG_TRANS_ADAPTER_EVENTS	The number of events dispatched to the Transaction EP adapter. Reset characteristic: reset to zero
Events to Tsqueue EP adapter	EPG_TSQ_ADAPTER_EVENT	The number of events dispatched to the TS queue EP adapter. Reset characteristic: reset to zero

Table 87. EVENTPROCESS: global statistics (continued)

DFHSTUP name	Field name	Description
Events to Custom EP adapter	EPG_CUSTOM_ADAPTER_EVENTS	The number of events dispatched to the Custom EP adapter. Reset characteristic: reset to zero
Events to HTTP EP adapter	EPG_HTTP_ADAPTER_EVENTS	The number of events dispatched to the HTTP EP adapter. Reset characteristic: reset to zero
Events lost (dispatch) - config	EPG_DISPATCH_FAILURE_CONFIG	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem relating to a resource specified in the `eventDispatcherPolicy` section of the event binding. Reset characteristic: reset to zero
Events lost (dispatch) - other	EPG_DISPATCH_FAILURE_OTHER	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem in the CICS environment, for example, insufficient storage. Reset characteristic: reset to zero
Events lost (adapter) - config	EPG_ADAPTER_FAILURE_CONFIG	The number of events that were captured but not emitted because the EP adapter encountered a problem relating to a resource specified in the `eventDispatcherAdapter` configuration section of the event binding. Reset characteristic: reset to zero
Events lost (adapter) - other	EPG_ADAPTER_FAILURE_OTHER	The number of events that were captured but not emitted because the EP adapter encountered a problem in the CICS environment, for example, insufficient storage. Reset characteristic: reset to zero
Events lost - adapter unavailable	EPG_EVENTS_ADAPTER_UNAVAIL	The number of events that were not emitted because the EP adapter is disabled or not installed. Reset characteristic: reset to zero

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTBINDING: Global statistics" on page 504
Shows information and global statistics about event bindings.

"EVENTBINDING: Summary global statistics" on page 506
Shows information and summary global statistics about event bindings.

"EVENTBINDING: Resource statistics" on page 505
Shows information and resource statistics about event bindings.

"EVENTBINDING: Summary resource statistics" on page 506
Shows information and summary resource statistics about event bindings.

Related information:

EXTRACT STATISTICS in Reference > System programming

EVENTPROCESS: Summary global statistics

Shows information and summary global statistics about event processing.

Summary statistics are not available online.

Table 88. EVENTPROCESS: summary global statistics

DFHSTUP name	Description
Number of put events	The number of events passed to the EP component for emission.
Number of commit forward events	The number of units of work that have been committed, and that included one or more asynchronous transactional events.
Number of commit backward events	The number of units of work that have been backed out, and that included one or more asynchronous transactional events.
Current event capture queue	The current number of events on the event capture queue.
Peak event capture queue	The peak number of events on the event capture queue.
Current transactional queue	The current number of events on the transactional queue.
Peak transactional queue	The peak number of events on the transactional queue.
Number of async normal events	The number of asynchronous normal priority events.
Number of async priority events	The number of asynchronous high priority events.
Number of transactional events	The number of transactional events.
Transactional events discarded	The number of transactional events discarded.
Number of synchronous events	The number of synchronous emission events captured.
Number of sync events failed	The number of synchronous emission events that were not emitted.
Number of dispatcher attaches	The number of dispatcher tasks attached.
Current dispatcher tasks	The current number of dispatcher tasks.
Peak dispatcher tasks	The peak number of dispatcher tasks.
Events to WebSphere MQ EP adapter	The number of events dispatched to the WebSphere MQ EP adapter.

Table 88. EVENTPROCESS: summary global statistics (continued)

DFHSTUP name	Description
Events to transaction EP adapter	The number of events dispatched to the Transaction EP adapter.
Events to Tsqueue EP adapter	The number of events dispatched to the TS queue EP adapter.
Events to custom EP adapter	The number of events dispatched to the Custom EP adapter.
Events to HTTP EP adapter	The number of events dispatched to the HTTP EP adapter.
Events lost (dispatch) - config	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem relating to a resource specified in the `eventDispatcherPolicy` section of the event binding.
Events lost (dispatch) - other	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem in the CICS environment, for example, insufficient storage.
Events lost (adapter) - config	The number of events that were captured but not emitted because the EP adapter encountered a problem relating to a resource specified in the `eventDispatcherAdapter` configuration section of the event binding.
Events lost (adapter) - other	The number of events that were captured but not emitted because the EP adapter encountered a problem in the CICS environment, for example, insufficient storage.
Events lost - adapter unavailable	The number of events that were not emitted because the EP adapter is disabled or not installed.

Related reference:

"CAPTURESPEC: Resource statistics" on page 499
You can retrieve CAPTURESPEC resource statistics by using the **EXEC CICS EXTRACT STATISTICS CAPTURESPEC RESID()** command. They are mapped by the DFHECCDS DSECT.

"CAPTURESPEC: Summary resource statistics" on page 500
Shows summary information and statistics about the capture specifications for each event.

"EVENTBINDING: Global statistics" on page 504
Shows information and global statistics about event bindings.

"EVENTBINDING: Summary global statistics" on page 506
Shows information and summary global statistics about event bindings.

"EVENTBINDING: Resource statistics" on page 505
Shows information and resource statistics about event bindings.

"EVENTBINDING: Summary resource statistics" on page 506
Shows information and summary resource statistics about event bindings.

Front end programming interface (FEPI) statistics

FEPI statistics contain data about the use of each FEPI connection, each FEPI pool, and a target in any pool.

CICS monitoring and statistics data can be used to help tune FEPI applications, and to control the resources that they use. For information about the performance aspects of the FEPI, see Improving FEPI performance in Improving performance.

FEPI: Connection statistics

You can retrieve FEPI connection statistics by using the **EXEC CICS COLLECT STATISTICS NODE TARGET** system command. They are mapped by the DFHA23DS DSECT.

Table 89. FEPI: Connection statistics

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 z/OS Communications Server error conditions raised for this connection. Reset characteristic: reset to zero

Related reference:

COLLECT STATISTICS in Reference > System programming

FEPI: Pool statistics

You can retrieve FEPI pool statistics by using the **EXEC CICS COLLECT STATISTICS POOL** system command. They are mapped by the DFHA22DS DSECT.

Table 90. FEPI: Pool statistics

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
Allocate Waits NOT IN THE DFHSTUP REPORT	A22WAIT	is the current number of conversations waiting to be allocated. Reset characteristic: not reset

Table 90. FEPI: Pool statistics (continued)

DFHSTUP name	Field name	Description
–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

Related reference:

COLLECT STATISTICS in Reference > System programming

FEPI: Target statistics

You can retrieve statistics a particular target in a pool by using the **EXEC CICS COLLECT STATISTICS POOL TARGET** system command. They are mapped by the DFHA24DS DSECT.

Table 91. FEPI: Target statistics

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 z/OS Communications Server 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		

Table 91. FEPI: Target statistics (continued)

DFHSTUP name	Field name	Description
–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

Related reference:

COLLECT STATISTICS in Reference > System programming

FEPI: Unsolicited connection statistics

Unsolicited connection statistics are produced when a connection is destroyed. This occurs when an **EXEC CICS FEPI 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 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 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: Summary connection statistics

FEPI Summary connection statistics are not available online.

Table 92. FEPI: Summary connection statistics

DFHSTUP name	Description
Pool name	is the FEPI pool name.
Target name	is the FEPI target name.
Node name	is the FEPI node.

Table 92. FEPI: Summary connection statistics (continued)

DFHSTUP name	Description
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 z/OS Communications Server error conditions raised for this connection.

FEPI: Summary pool statistics

FEPI Summary pool statistics are not available online.

Table 93. FEPI: Summary pool statistics

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: Summary target statistics

FEPI: Summary target statistics are not available online.

Table 94. FEPI: Summary target statistics

DFHSTUP name	Description
Target name	is the FEPI target name.
Pool name	is the FEPI pool name.
Applid	is the z/OS Communications Server 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	
–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 control statistics

There are four sections in the DFHSTUP report for file statistics, dealing with resource information, requests information, data table requests information, and performance information.

Unsolicited file statistics are printed in a statistics report separate from other types of CICS statistics.

You can retrieve these statistics by using the **EXEC CICS EXTRACT STATISTICS FILE** system command. They are mapped by the DFHA17DS DSECT.

Related reference:

"Files report" on page 834
The Files report is produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

"File Requests report" on page 835
The File Requests report is produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

"Data Tables reports" on page 804
The Data Tables Requests and Data Tables Storage reports are produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

"Data Set Name report" on page 804
The Data Set Name report is produced using the **EXEC CICS INQUIRE DSNAME** command.

Related information:

EXTRACT STATISTICS in Reference > System programming

Interpreting 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.

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 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 “Interpreting LSR pool statistics” on page 618) 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.

Files: Resource statistics - resource information

The file resource information statistics provide information about files.

Table 95. Files: Resource statistics - resource information

DFHSTUP name	Field name	Description
File name	A17FNAM	The name you specified in the DEFINE FILE command of resource definition online. Reset characteristic: not reset

Table 95. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Data set name	A17DSNAM	The 44-character name that defines the physical data set to the system. This name can be specified as follows: • 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 • 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
Base data set name (if applicable)	A17BDSNM	If the file is a VSAM PATH, this field gives the base data set name. Reset characteristic: not reset.
Data set type	A17DSTYP	The data set type, which can be BDAM, standard ESDS, extended ESDS, KSDS, RRDS, VRRDS, or PATH. If the file is remote or not open, this field is blank. **Key** **Statistics type** **B** BDAM **E** Standard ESDS **K** KSDS **P** PATH **R** RRDS **V** VRRDS **X** Extended ESDS Reset characteristic: not reset.
RLS	A17DSRLS	Indicates whether the file is RLS. • 'R' =RLS accessed file • ' ' =Non-RLS These values are shown as Yes and No, respectively, in the DFHSTUP report. Reset characteristic: not reset.

Table 95. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
DataTable indicator	A17DT	A 1-byte field that contains the value R, S T, L K, 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 shared data table. • L indicates that the resource is a coupling facility data table (locking model). • K indicates that the resource is a coupling facility data table (contention model). • X indicates that the resource has been opened with a source data set which has an associated CICS maintained data table and the resource has been updated which has caused the data table to also be updated. Reset characteristic: not reset
Time opened	A17LOPNT	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	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

Table 95. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
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 IPIC, 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.
LSR	A17POOL	The identity of the local shared resource pool. This value is that specified by: • The LSRPOOLNUM operand of the resource definition online DEFINE FILE command. "N" means that it is not defined in an LSR pool. Reset characteristic: not reset.
CFDT PoolName	A17DTCFP	The name of the coupling facility data table pool defined for the data table associated with the file 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
Not in DFHSTUP report	A17_FILE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset

Table 95. Files: Resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	A17_FILE_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_USERID	The user ID that installed the resource. 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.

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Files: Resource statistics - requests information

File requests information statistics provide information about the requests made against files.

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.

Table 96. Files: Resource statistics - requests information

DFHSTUP name	Field name	Description
File name	A17FNAM	is the name you specified in: • The DEFINE FILE command of resource definition online • (for BDAM files only) 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	is the number of DELETE requests attempted against this file. Reset characteristic: reset to zero
Brws upd requests	A17DSBRU	is the number of browse READNEXT UPDATE and READPREV UPDATE requests issued against this file. Note that this field is only applicable to RLS accessed files. Reset characteristic: reset to zero
VSAM EXCP requests		

Table 96. Files: Resource statistics - requests information (continued)

DFHSTUP name	Field name	Description
–Data	A17DSXCP	A value is printed if the file has been opened and used as a VSAM KSDS during the CICS run, even if the file is not being used as a KSDS at the time of taking statistics. See notes 1 on page 525, 2 on page 525 and 3 on page 525.
–Index	A17DSIXP	See notes 1 on page 525, 2 on page 525 and 3 on page 525. Reset characteristic: reset to zero
RLS req timeouts	A17RLSWT	is the number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated. Reset characteristic: reset to zero
Not in DFHSTUP report	A17_FILE_DEFINE_SOURCE	The name of the CSD group that contains to this resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

Table 96. Files: Resource statistics - requests information (continued)

DFHSTUP name	Field name	Description

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 file. If dynamic allocation has been used to change the physical data sets associated with a file, 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. When the count of EXCPs rises above x'80000000', the count is no longer reliable, and you should consider redefining the file.
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.
4. The EXCP count for RLS files is the count of all EXCPs for all tasks accessing the RLS file within that CICS region. It should be noted as stated in note 2, EXCP counts are stored in the file's corresponding ACB within that CICS region.

Files: Resource statistics - data table requests information

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 "L", or "K", or "X" in the field A17DT. Their names and meanings are as follows:

Table 97. Files: Resource statistics - data table requests information

DFHSTUP name	Field name	Description
File Name	A17FNAM	The name you specified in the DEFINE FILE command of resource definition online. Reset characteristic: not reset
Close type	A17DTTYP	This 1 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-maintained table is being closed • "U" when a user maintained table is closed • "L" when a locking model coupling facility data table is closed • "K" when a contention model coupling facility data table is closed. Reset characteristic: not reset

Table 97. Files: Resource statistics - data table requests information (continued)

DFHSTUP name	Field name	Description
Read requests	A17DTRDS	The number of attempts to retrieve records from the table. Reset characteristic: reset to zero
Recs¬[not] in table	A17DTRNF	The number of reads where the record was not found in the data table, so CICS retrieved the record from the source file. Reset characteristic: reset to zero
Adds from reads	A17DTAVR	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	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	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	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	The number of attempts to update records in the table as a result of REWRITE requests. Reset characteristic: reset to zero
Delete requests	A17DTDLS	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	The peak number of records present in the table. Reset characteristic: reset at close

Table 97. Files: Resource statistics - data table requests information (continued)

DFHSTUP name	Field name	Description
Storage alloc(K)	A17DTALT	The total amount of storage allocated to the data table. The DFHSTUP report expresses the storage in KB. DFHSTUP does not total the storage allocated for all data tables because multiple files can share the same data table. Reset characteristic: not reset
Chng Resp/Lock Waits	A17DTCON	For a CFDT that is using the locking model, records are locked down when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record. For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed, a CHANGED response is returned. This count is the number of times that a CHANGED response was issued. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	A17DTLDS	The number of times that a LOADING response was issued. When a CFDT is in the process of being loaded, and requests issued for records beyond the range of those already loaded get a LOADING response. Reset characteristic. reset to zero

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, updating activity is not shown in the data table resource information.

When using the shared data tables feature the statistics records contain the additional information as follows:

Table 98. Files: shared data table statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A17DTSIZ	The current number of records in the data table. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUST	The total amount of storage (KB) in use for the data table. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALE	The total amount of storage (KB) allocated for the record entry blocks. Reset characteristic: not reset

Table 98. Files: shared data table statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A17DTUSE	The total amount of storage (KB) in use for the record entry blocks. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALI	The total amount of storage (KB) allocated for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSI	The total amount of storage (KB) in use for the index. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTALD	The total amount of storage (KB) allocated for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTUSD	The total amount of storage (KB) in use for the record data. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A17DTRRS	The total number of read retries, that is the number of times reads in an AOR must 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 wanted to read. In such cases, the request is function shipped and is counted in the “source reads”. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_DEFINE_SOURCE	The name of the CSD group that contains to this resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset

Table 98. Files: shared data table statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	A17_FILE_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_USERID	The user ID that installed the resource. 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: Resource statistics - performance information

These statistics are available online, and are mapped by the DFHA17DS DSECT.

Table 99. Files: Resource statistics - performance information

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
Strings	A17STRNO	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed. Reset characteristic: not reset.
Active strings	A17DSASC	The current number of updates against the file. Reset characteristic: not reset.
Wait on Strings: Current	A17DSASW	The current number of 'waits' for strings against the file. Reset characteristic: not reset

Table 99. Files: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Wait on Strings: Total	A17DSTSW	The total number of 'waits' for strings against the file. Reset characteristic: reset to zero
Wait on Strings: Highest	A17DSHSW	The highest number of 'waits' for strings against the file. Reset characteristic: reset to current value
Buffers: Data	A17DSDNB	is the number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for z/OS UNIX files. Reset characteristic: not reset.
Buffers: Index	A17DSINB	is the number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for z/OS UNIX files. Reset characteristic: not reset.
Excl Cntl Conflicts	A17FCXCC	is the number of exclusive control conflicts that have occurred against VSAM control intervals in this file. Reset characteristic: reset to zero
Not in DFHSTUP report	A17_FILE_DEFINE_SOURCE	The name of the CSD group that contains to this resource. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset

Table 99. Files: Resource statistics - performance information (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	A17_FILE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	A17_FILE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

Files: Summary statistics - resource information

File resource summary statistics are unavailable online.

Table 100. Files: Summary statistics - resource information

DFHSTUP name	Description
File Name	The name you specified in the DEFINE FILE command of resource definition online.
Data set name	The 44-character name defining the physical data set to the system. For remote files the data set name is shown as REMOTE.
Base data set name (If applicable)	In the instance that the file is a VSAM PATH, this field gives the base data set name.
Data set type	The data set type, which can be BDAM, standard ESDS, extended ESDS, KSDS, RRDS, VRRDS, or PATH. If the file is remote or not open, this field is blank. **Key** Statistics type **B** BDAM **E** Standard ESDS **K** KSDS **P** PATH **R** RRDS **V** VRRDS **X** Extended ESDS
RLS	An indicator of whether the file is RLS accessed or not. YES indicates an RLS-accessed file; NO indicates a non-RLS file.

Table 100. Files: Summary statistics - resource information (continued)

DFHSTUP name	Description
Data Table indicator	A 1-byte field that contains one of the following values: R, S, T, L, K, 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. • L indicates that the resource is a coupling facility data table that uses the locking model. • K indicates that the resource is a coupling facility data table that uses the contention model. • X indicates that the resource has been opened with a source data set that has an associated CICS maintained data table, and the resource has been updated, which has caused the data table to also be updated.
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 IPIC, ISC, or MRO environment, and the file is held by a remote system, this field specifies the system upon which the file is resident.
LSR	The identity of the local shared resource pool. This value is that specified using the LSRPOOLNUM operand of the resource definition online DEFINE FILE command."N" means that it is not defined in an LSR pool.
CFDT PoolName	The name of the coupling facility data table pool defined for the data table associated with the file.

Files: Summary statistics - requests information

File requests summary statistics are not available online.

Table 101. Files: Summary statistics - requests information

DFHSTUP name	Description
File name	is the name you specified in: • The DEFINE FILE command of resource definition online • (for BDAM files only) 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.

Table 101. Files: Summary statistics - requests information (continued)

DFHSTUP name	Description
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.
Brws upd requests	is the total number of READNEXT UPDATE and READPREV UPDATE requests issued against this file (RLS only).
VSAM EXCP request: Data	A value is printed if the file has been opened and used as a VSAM KSDS during the CICS run. See notes 1, 2 and 3.
VSAM EXCP request: Index	See notes 1, 2 and 3.
VSAM EXCP request: RLS req timeouts	is the total number of RLS requests made to this file that were not serviced in the specified time limit, and therefore the requests were terminated.

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 file. If dynamic allocation has been used to change the physical data sets associated with a file, 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. When the count of EXCPs rises above x'80000000', the count is no longer reliable, and you should consider redefining the file.
3. For RLS, this value is a count of the number of calls to the system buffer manager. It includes calls that result in either a coupling facility cache access or an I/O.
4. The EXCP count for RLS files is the count of all EXCPs for all tasks accessing the RLS file within that CICS region. It should be noted as stated in note 2, EXCP counts are stored in the file's corresponding ACB within that CICS region.

Files: Summary statistics - data table requests information

File data table requests summary statistics are unavailable online.

Table 102. Files: Summary statistics - data table requests information

DFHSTUP name	Description
File Name	The name you specified in the DEFINE FILE command of resource definition online.

Table 102. Files: Summary statistics - data table requests information (continued)

DFHSTUP name	Description
Table type	This 1 byte field is set as follows: • C when a CICS maintained table is closed. • P when a file that accessed a CICS maintained table is closed, but the table remains open because other files that are using the table are still open. • S when the source data set for a user maintained table is being closed. • U when a user maintained table is closed. • L when a locking model coupling facility data table is closed. • K when a contention model coupling facility data table is closed.
Successful reads	The total number of reads from the data table.
Recs in table	The number of reads where the record was not found in the data table, so CICS retrieved the record from the source file.
Adds from reads	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	The total number of attempts to add records to the table as a result of WRITE requests.
Adds rejected - exit	The total number of records CICS attempted to add to the table that were rejected by the global user exit.
Adds rejected - table full	The total number of records CICS attempted to add to the table but was could not because the table already contained the maximum number of records specified.
Rewrite requests	The total number of attempts to update records in the table as a result of REWRITE requests.
Delete requests	The total number of attempts to delete records from the table as a result of DELETE requests.
Highest table size	The peak number of records present in the table.
Chng Resp/Lock Waits	For a CFDT that uses the locking model, records are locked down when they are read for update. This count is the number of times it was necessary to wait for a record that was already locked. For a CFDT that uses the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed, a CHANGED response is returned. This count is the number of times that a CHANGED response was issued.

Files: Summary statistics - performance information

File performance summary statistics are unavailable online.

Table 103. Files: Summary statistics - performance information

DFHSTUP name	Description
File name	The name you specified in the DEFINE FILE command of resource definition online.
Strings	The maximum permissible number of concurrent updates. For RLS, the value specified in the ACB macro is ignored. After OPEN a value of 1024 is returned, indicating the maximum number of strings allowed.
Wait on strings: Total	The total number of 'waits' for strings against the file.
Wait on strings: HWM	The highest number of 'waits' for strings against the file.

Table 103. Files: Summary statistics - performance information (continued)

DFHSTUP name	Description
Buffers: Data	The number of buffers to be used for data. For RLS, BUFND is ignored and the value specified in the ACB is returned. This parameter has no effect for z/OS UNIX files.
Buffers: Index	The number of buffers to be used for index. For RLS, BUFNI is ignored and the value specified in the ACB is returned. This parameter has no effect for z/OS UNIX files.
Excl Cntl Conflicts	The total number of exclusive control conflicts that have occurred against VSAM control intervals in this file.

ISC/IRC system and mode entry statistics

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.

Note: ISC/IRC system and mode entry statistics contain information about intersystem communication over SNA (ISC over SNA) and multiregion operation (MRO) connections. Information about IP interconnectivity (IPIC) connections is in IPCONN statistics.

The two types of intersystem communication, ISC over SNA and IPIC, are described in Intersystem communication, in the *CICS Intercommunication Guide*.

Related concepts:

"Interpreting 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.

Related reference:

"Connections and Modenames report" on page 799
The Connections and Modenames report is produced using a combination of the **EXEC CICS INQUIRE CONNECTION**, **EXEC CICS INQUIRE MODENAME** and **EXEC CICS COLLECT STATISTICS CONNECTION** commands. The statistics data is mapped by the DFHA14DS DSECT.

Interpreting 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 topics identify the questions you might have about system performance, and describe how answers to those questions can be derived from the statistics report. The topics also describe what actions, if any, you can take to resolve ISC/IRC performance problems.

Here are some questions you might have:
- 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 104. 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 in the table are specific to the mode group of the mode name given.

Table 105. 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 105. ISC/IRC mode entries (continued)

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 entry statistics" on page 535.

General guidance for interpreting ISC/IRC statistics

Guidance information on interpreting the ISC/IRC statistics

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 that 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 104 on page 536 and Table 105 on page 536.
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, 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 the network.
     - Increased use of storage by z/OS Communications Server.
     - 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 slightly more than 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, because 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
   - Anticipate potential performance problems before they become critical.

Are 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.

The peak fields 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, the total number of sessions defined is too low.

3. *"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.

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

   If the number of "Peak contention losers" is equal to the number of contention loser sessions available, 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 details of how to specify the maximum number of sessions, and the number of contention winners, see the information on defining SESSIONS in SESSION resource definitions in the *CICS 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 59 for an illustration of this. To change the order of the TCTMEs, you must change the names of the SESSIONS definitions. You can 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 further down the TCTME chain, and avoid conflict with the generic ALLOCATEs. *Alternatively, make all allocates specific allocates*.

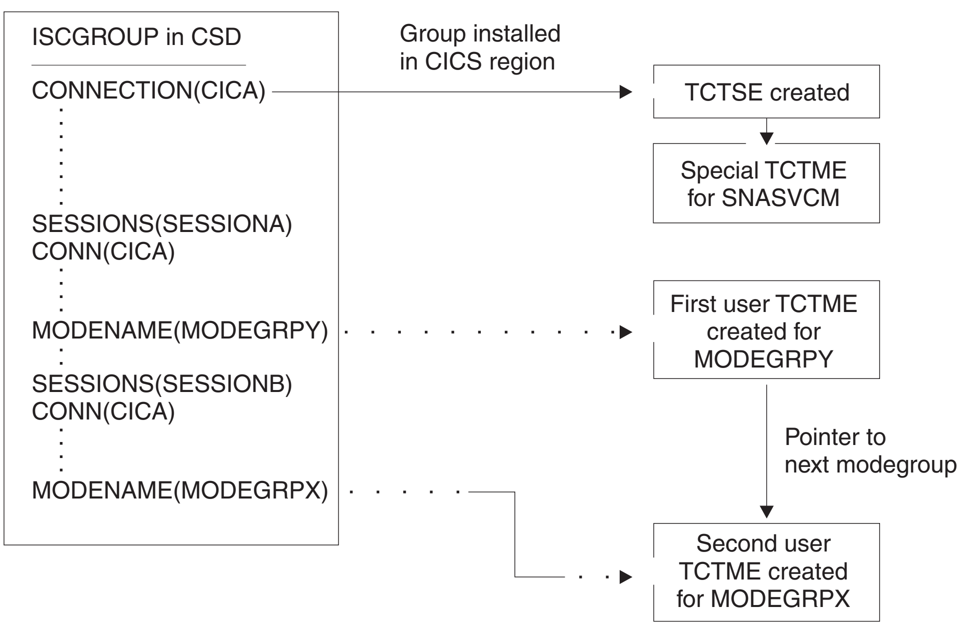


Figure 59. How the sequence of TCT mode entries is determined

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, 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. If IRC, increase the RECEIVES.

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 that the connection is released, out of service, or 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 reacquire a released connection. This may help to minimize outage time caused by connections being unavailable for use. See INQUIRE CONNECTION, INQUIRE MODENAME, SET CONNECTION, and SET MODENAME in the *CICS System Programming Reference* for programming information about the 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. Increase the sessions if IRC is used.

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 system entry: Resource statistics

You can retrieve ISC/IRC system entry resource statistics by using the **EXEC CICS EXTRACT STATISTICS CONNECTION** system command. They are mapped by the DFHA14DS DSECT,

The system entry statistics record information for both ISC and IRC connections. Some of the information is unique to each type of connection. ISC/IRC system and mode entry statistics contain information about intersystem communication over SNA (ISC over SNA) and multiregion operation (MRO) connections. Information about IP interconnectivity connections is in IPCONN statistics.

Note:

The two types of intersystem communication, ISC over SNA and IPIC, are described in Intersystem communication, in the *CICS Intercommunication Guide*.

This DSECT is to be used:

- For processing data returned for an online enquiry for a connection (**EXEC CICS EXTRACT 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.

Table 106. ISC/IRC system entry: Resource statistics

DFHSTUP name	Field name	Description
Connection name	A14CNTN	corresponds to each system entry defined by a CONNECTION definition in the CSD, or by autoinstall. Reset characteristic: not reset
Connection netname	A14ESID	is the name by which the remote system is known in the network—that is, its applid. Reset characteristic: not reset

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Access Method / Protocol	A14ACCM	is the communication access method used for this connection. The values are: • X'01' =A14VTAM • X'02' =A14IRC • X'03' =A14XM • X'04' =A14XCF
Access Method / Protocol	A14EFLGS	is the communication protocol used for this connection. The values are: • X'01' =A14APPC • X'02' =A14LU61 • X'03' =A14EXCI Reset characteristic: not reset
Autoinstalled Connection Create Time	A14AICT	is the time at which this connection was autoinstalled, in local time. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00').
Autoinstalled Connection Delete Time	A14AIDT	is the time at which this connection was deleted, in local time. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00').
Send session count	A14ESECN	is the number of SEND sessions for this connection. This field applies to MRO and LU6.1 connections only. Reset characteristic: not reset
Receive session count	A14EPRMN	is the number of RECEIVE sessions for this connection. This field applies to MRO and LU6.1 connections only. 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

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
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 EXTRACT 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 EXTRACT STATISTICS** command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
Current contention losers	A14E1RY	is the number of contention loser sessions (primaries for LU6.1) that are currently in use. Reset characteristic: not reset
Peak contention losers	A14E1HWM	is the peak number of contention loser sessions (primaries for LU6.1) that were in use at any one time. Reset characteristic: reset to current value
Current contention winners	A14E2RY	is the number of contention winner sessions (secondaries for LU6.1) that are currently in use. Reset characteristic: not reset
Peak contention winners	A14E2HWM	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time. Reset characteristic: reset to current value

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
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 EXTRACT STATISTICS** command, this field is the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
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 EXTRACT 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
Peak outstanding allocates For more information see note following this table.	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
Total number of allocates For more information see note following this table.	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 EXTRACT STATISTICS** command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Queued allocates For more information see note following this table.	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 EXTRACT STATISTICS** command, this field also contains the summation of the equivalent mode entry statistics. Reset characteristic: not reset
Failed link allocates For more information see note following this table.	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 EXTRACT 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 For more information see note following this table.	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 EXTRACT 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

Table 106. ISC/IRC system entry: Resource statistics (continued)

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. If a QUEUELIMIT of No has been set, this field has a value of -1. 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 EXTRACT STATISTICS** command, this field additionally contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
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 EXTRACT STATISTICS** command, this field additionally contains the summation of the equivalent mode entry statistics. Reset characteristic: reset to zero
File control (FC) function shipping requests	A14ESTFC	is the number of file control requests for function shipping. Reset characteristic: reset to zero
Interval control IC) function shipping requests	A14ESTIC	is the number of interval control requests for function shipping. Reset characteristic: reset to zero
Program control (PC) function shipping requests	A14ESTPC	is the number of program control link requests for function shipping. Reset characteristic: reset to zero
Transient data (TD) function shipping requests	A14ESTTD	is the number of transient data requests for function shipping. Reset characteristic: reset to zero
Temporary storage (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

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	A14GACT	is the time at which this connection was autoinstalled, in GMT. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only applicable to an autoinstalled APPC connection. For all other types of connection the value will be nulls (x'00'). Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A14GADT	is the time at which this connection was deleted, in GMT. The time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK). This field is only set if this is an autoinstalled APPC connection that has been deleted, that is, this field is only set in an unsolicited statistics (USS) record. For all other types of connection and all other types of statistics record the value will be nulls (x'00'). Reset characteristic: not reset
Terminal-sharing channel requests	A14ESTTC_CHANNEL	is the number of terminal-sharing channel requests. Reset characteristic: reset to zero
Number of bytes sent on terminal-sharing channel requests	A14ESTTC_CHANNEL_SENT	is the number of bytes sent on terminal-sharing channel requests. This is the total amount of data sent on the connection, including any control information. Reset characteristic: reset to zero
Number of bytes received on terminal-sharing channel requests	A14ESTTC_CHANNEL_RCVD	is the number of bytes received on terminal-sharing channel requests. This is the total amount of data sent on the connection, including any control information. Reset characteristic: reset to zero
Program control function-shipping LINK requests, with channels	A14ESTPC_CHANNEL	is the number of program control LINK requests, with channels, for function shipping. This is a subset of the number in A14ESTPC. Reset characteristic: reset to zero
Number of bytes sent on LINK channel requests	A14ESTPC_CHANNEL _SENT	is the number of bytes sent on LINK channel requests. This is the total amount of data sent on the connection, including any control information. Reset characteristic: reset to zero

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Number of bytes received on LINK channel requests	A14ESTPC_CHANNEL _RCVD	is the number of bytes received on LINK channel requests. This is the total amount of data received on the connection, including any control information. Reset characteristic: reset to zero
Interval control function-shipping START requests, with channels	A14ESTIC_CHANNEL	is the number of interval control START requests, with channels, for function shipping. This is a subset of the number in A14ESTIC. Reset characteristic: reset to zero
Number of bytes sent on START channel requests	A14ESTIC_CHANNEL_SENT	is the number of bytes sent on START channel requests. This is the total amount of data sent on the connection, including any control information. Reset characteristic: reset to zero
Number of bytes received on START channel requests	A14ESTIC_CHANNEL_RCVD	is the number of bytes received on START channel requests. This is the total amount of data sent on the connection including any control information. Reset characteristic: reset to zero
Not in DFHSTUP report	A14ESTPC_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see The resource signature table. Reset characteristic: not reset
Not in DFHSTUP report	A14ESTPC_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	A14ESTPC_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	A14ESTPC_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	A14ESTPC_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset

Table 106. ISC/IRC system entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	A14ESTPC_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	A14ESTPC_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

Note:

1. For APPC only, if an allocate request does not specify a mode group (so it is a generic allocate request), CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry and against the mode entry (in the statistic 'Total generic allocates satisfied'). If an allocate specifically requests a mode entry (so it is a specific allocate request), the statistics for these allocates go into that mode entry.

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

ISC/IRC system entry: Summary resource statistics

ISC/IRC system entry summary resource statistics are not available online.

Table 107. ISC/IRC system entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the system entry defined by the CONNECTION definition in the CSD or by autoinstall.
Connection netname	is the name by which the remote system is known in the network—that is, its applid.
Access Method / Protocol	is the combined communication access method and protocol used for the connection.
Average autoinstalled connection time	is the average autoinstalled connection time. This field applies to autoinstalled connections and is summarized from the unsolicited system entry statistics records only.

Table 107. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Send session count	is the last value encountered for the SENDCOUNT specified on the CONNECTION definition. This field applies to MRO and LU6.1 connections only.
Receive session count	is the last value encountered for the RECEIVECOUNT specified on the CONNECTION definition. This field applies to MRO, LU6.1, and EXCI connections only.
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.
Peak contention winners	is the peak number of contention winner sessions (secondaries for LU6.1) that were in use at any one time.
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.
Peak outstanding allocates For more information see 1 on page 554	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 For more information see 1 on page 554	is the total number of allocate requests against this system. For APPC this field contains only generic allocate requests.
Average number of queued allocates For more information see 1 on page 554	is the average number of queued allocate requests against this system. For APPC this field is incremented only for generic allocate requests.

Table 107. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Failed link allocates For more information see 1 on page 554	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 For more information see 1 on page 554	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 parameter 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.
File control (FC) function shipping requests	is the total number of file control requests for function shipping.

Table 107. ISC/IRC system entry: Summary resource statistics (continued)

DFHSTUP name	Description
Interval control (IC) function shipping requests	is the total number of interval control requests for function shipping.
Program control (PC) function shipping requests	is the total number of program control link requests for function shipping.
Transient data (TD) function shipping requests	is the total number of transient data requests for function shipping.
Temporary storage (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.

Note:

1. For APPC only, if an allocate request does not specify a mode group (so it is a generic allocate request), CICS takes the first mode group within the sessions available, and the statistics for these allocates are reported against the system entry and against the mode entry (in the statistic 'Total generic allocates satisfied'). If an allocate specifically requests a mode entry (so it is a specific allocate request), the statistics for these allocates go into that mode entry.

ISC mode entry: Resource statistics

These statistics cannot be retrieved using the **EXEC CICS EXTRACT STATISTICS** command. They are only produced for offline processing (written to SMF).

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 are mapped by the DFHA20DS DSECT. This DSECT is also used to map the mode entry totals records.

Table 108. ISC mode entry: Resource statistics

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 or by autoinstall. Reset characteristic: not reset

Table 108. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Mode name	A20MODE	is the mode group name related to the the intersystem connection name (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
Current contention losers in use	A20E1RY	is the number of contention loser sessions currently in use. Reset characteristic: not reset
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) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time. Reset characteristic: reset to current value
Current contention winners in use	A20E2RY	is the number of contention winner sessions currently in use. Reset characteristic: not reset
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) as “contention winners” or “contention losers”, and their states are dynamically decided at bind time. Reset characteristic: reset to current value
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

Table 108. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
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
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
Peak outstanding allocates For more information see 1 on page 558	A20ESTAM	is the peak number of allocation requests that were queued for this mode group. Reset characteristic: reset to current value
Total specific allocate requests For more information see 1 on page 558	A20ESTAS	is the number of specific allocate requests against this mode group. Reset characteristic: reset to zero
Total specific allocates satisfied For more information see 1 on page 558	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
Queued allocates For more information see 1 on page 558	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 For more information see 1 on page 558	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

Table 108. ISC mode entry: Resource statistics (continued)

DFHSTUP name	Field name	Description
Failed allocates due to sessions in use For more information see 1 on page 558	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
Maximum session count	A20ELMAX	is the maximum number of sessions that the definition of the session group permits. Reset characteristic: not reset
Current maximum session count	A20EMAXS	is the current number of sessions in the group (the number "bound"). Reset characteristic: not reset
Maximum contention winners acceptable	A20EMCON	is the maximum number of sessions that the definition of the session group permits to be contention winners. Reset characteristic: not reset
Current CNOS contention losers	A20ECONL	is the current number of CNOS negotiated contention loser sessions. Reset characteristic: not reset
Current CNOS contention winners	A20ECONW	is the current number of CNOS negotiated contention winner sessions. Reset characteristic: not reset

Note:

1. This field is 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.

ISC mode entry: Summary resource statistics

ISC mode entry summary resource 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.

Table 109. ISC mode entry: Summary resource statistics

DFHSTUP name	Description
Connection name	is the name of the APPC connection/system that owns this mode entry.
Mode name	is the mode group name related to the intersystem connection name in the previous row. 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 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 as “contention winners” or “contention losers”, and their states are dynamically decided at bind time.
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.
Peak outstanding allocates	is the peak number of allocation requests that were queued for this mode group.
For more information see 1 on page 559	

Table 109. ISC mode entry: Summary resource statistics (continued)

DFHSTUP name	Description
Total specific allocate requests For more information see 1	is the total number of specific allocate requests against this mode group.
Total specific allocates satisfied For more information see 1	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 allocates are made for APPC without the mode group being specified.
Average number of queued allocates For more information see 1	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 For more information see 1	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 For more information see 1	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.
Number of XZIQUE allocates purged	is the total number of allocates purged due to XZIQUE requesting that queues should be purged (Number of XZIQUE allocate queue purges) 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.

Note:

1. The next three fields only contain allocates against specific mode groups. Generic allocate requests are contained in the equivalent system entry statistics.

ISC/IRC attach time entry statistics

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 Persistent Verification “signed on from” list are either reused or timed out. Using this data you can adjust the USRDELAY, and the PVDELAY system initialization parameters.

Related concepts:

“Interpreting ISC and IRC attach time entry statistics” on page 560
ISC and IRC signon activity and ISC persistent verification (PV) activity give information about the best settings for your **USRDELAY** and **PVDELAY** system initialization parameters.

Interpreting ISC and IRC attach time entry statistics

ISC and IRC signon activity and ISC persistent verification (PV) activity give information about the best settings for your **USRDELAY** and **PVDELAY** system initialization parameters.

If the number of entries reused in signon activity is low, and the entries timed out value for signon activity is high, increase the value of the **USRDELAY** system initialization parameter. The average reuse time between entries value gives some indication of the time that you might want to set for the **USRDELAY** system initialization parameter.

Review your **USRDELAY** system initialization parameter, because with z/OS 1.11 and later, CICS is notified immediately if RACF profile changes occur.

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, increase the **PVDELAY** system initialization parameter. The average reuse time between entries value gives some indication of the time that you might want to set for the **PVDELAY** system initialization parameter.

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

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 retrieved using the **EXEC CICS EXTRACT STATISTICS** command; they are only produced for offline processing (written to SMF).

These statistics are mapped by the DFHA21DS DSECT.

Table 110. ISC/IRC attach time: Resource statistics

DFHSTUP name	Field name	Description
Persistent Verification refresh time	A21_SIT_LUIT_TIME	is the time in minutes set by the PVDELAY system initialization parameter. It specifies the password re-verification interval. The range is from zero through 10080 minutes (seven days) and the default is 30 minutes. If a value of zero is specified, 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

Table 110. ISC/IRC attach time: Resource statistics (continued)

DFHSTUP name	Field name	Description
ISC Persistent Verification Activity: 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
ISC Verification Activity: 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

ISC/IRC attach time: Summary resource 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.

Table 111. ISC/IRC attach time: Summary resource statistics

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.
Entries reused	refers to the number of times that user's entries in the PV 'signed on from' list were reused without referencing the ESM of the remote system.
Entries timed out	refers to the number of user's entries in the PV 'signed on from' list that were timed out after a period of inactivity.
Average reuse time between entries	refers to the average amount of time that has elapsed between each reuse of a user's entry in the PV 'signed on from' list.

IPCONN statistics

You can use IPCONN statistics to detect problems with IPIC connections.

IPIC is described in Intersystem communication, in the *CICS Intercommunication Guide*.

Interpreting IPCONN statistics

Information about the purpose of IPCONN statistics.

Note: Information about intersystem communication over SNA (ISC over SNA) and MRO connections is in ISC/IRC system and mode entry statistics.

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

- Are there enough sessions defined?
- Is the balance of receive and send sessions correct?
- What can be done if there are unusually high numbers, compared with normal or expected numbers, in the statistics report?

IPCONN: Resource statistics

You can retrieve IPCONN statistics by using the **EXEC CICS EXTRACT STATISTICS** system command. They are mapped by the DFHISRDS DSECT.

IPCONN statistics

You can use IPCONN statistics to detect problems with IP interconnectivity (IPIC) connections.

IPIC is described in Communication between systems in Getting started.

Use the DFHISRDS DSECT to process the following information:

- Data returned for an online enquiry for a connection (EXEC CICS EXTRACT STATISTICS)
- Connection statistics offline (SMF)
- Connection totals (the summation of all defined connections in this CICS region).

Table 112. IPCONN: resource statistics

DFHSTUP name	Field name	Description
IPCONN Name	ISR_IPCONN_NAME	The name of an IPIC connection defined by an IPCONN definition in the CSD or by autoinstall. Reset characteristic: not reset
Autoinstalled IPCONN Create Date / Time	ISR_IPCONN_CREATE_TIME	The date and time when the IPCONN was autoinstalled. The time shown is local time. If the IPCONN was not autoinstalled, this field is not shown.
Autoinstalled IPCONN Delete Date / Time	ISR_IPCONN_DELETE_TIME	The date and time when the autoinstalled IPCONN was deleted. The time shown is local time. If the IPCONN was not autoinstalled, this field is not shown.
IPCONN Applid	ISR_APPLID	The APPLID of the remote system, as specified in its system initialization table. Reset characteristic: not reset

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
IPCONN Network ID	ISR_NETWORK_ID	The network ID (that is, the z/OS Communications Server NETID or, for non-z/OS Communications Server systems, the value of the UOWNETQL system initialization parameter) of the remote system. This ID is used, in combination with the APPLID, to ensure unique naming for connecting systems. The name can be up to 8 characters in length and follows assembler language rules. It must start with an alphabetic character. This attribute is optional. If not specified, the z/OS Communications Server NETID (or, for non-z/OS Communications Server systems, the value of the UOWNETQL system initialization parameter) of the CICS on which the definition is installed is used. Reset characteristic: not reset
TCPIPSERVICE Name	ISR_TCPIP_SERVICE	The name of the PROTOCOL(IPIC) TCPIPSERVICE definition that defines the attributes of the inbound processing for this connection.
IPCONN Port Number	ISR_PORT_NUMBER	The decimal number of the port that is combined with the HOST value to specify the destination for outbound requests on this connection. Reset characteristic: not reset
IPCONN Host	ISR_HOST_NAME	The host name of the target system for this connection. Reset characteristic: not reset
IPCONN IP Family	ISR_IPCONN_IP_FAMILY	The address format of the IP Resolved Address. Reset characteristic: not reset
IPCONN IP Resolved Address	ISR_IPCONN_IP_ADDRESS	The IPv4 or IPv6 address of the host. Reset characteristic: not reset

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Receive Sessions	ISR_RECEIVE_SESSIONS	The defined number of receive sessions. The actual number of receive sessions that are used depends also on the number of send sessions defined in the remote system. When the connection is established, these values are exchanged and the lower value is used. Reset characteristic: not reset
Current Receive Sessions	ISR_CURRENT_RECEIVE_SESSIONS	The current number of receive sessions in use for this connection. Reset characteristic: reset to current value
Peak Receive Sessions	ISR_PEAK_RECEIVE_SESSIONS	The peak number of receive sessions in use for this connection. Reset characteristic: reset to current value
Total Allocates	ISR_TOTAL_ALLOCATES	The total number of allocate requests for this connection. Reset characteristic: reset to zero
Current Allocates Queued	ISR_CURRENT_QUEUED_ALLOCATES	The current number of allocate requests that have been queued for this connection. Reset characteristic: reset to current value
Peak Allocates Queued	ISR_PEAK_QUEUED_ALLOCATES	The peak number of allocate requests that have been queued for this connection. Reset characteristic: reset to current value
Allocates Failed - Link	ISR_ALLOCATES_FAILED_LINK	The number of allocate requests that failed because the connection is released or out-of-service. Reset characteristic: reset to zero
Allocate queue limit	ISR_ALLOCATE_QUEUE_LIMIT	The value of the QUEUELIMIT parameter specified on the IPCONN definition. This value is the maximum number of allocate requests that CICS is to queue while waiting for free sessions.

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Maximum queue time (seconds)	ISR_MAX_QUEUE_TIME	The MAXQTIME specified on the IPCONN definition. This value represents the maximum time that queued allocate requests, waiting for free sessions on a connection that appears to be unresponsive, can wait. The maximum queue time is used only if a queue limit is specified for QUEUELIMIT; and the time limit is applied only when the queue length has reached the queue limit value. Reset characteristic: not reset
Number of MAXQTIME allocate queue purges	ISR_MAXQTIME_ALLOC_QPURGES	The total number of times an allocate queue has been purged because of the MAXQTIME value. 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	ISR_MAXQTIME_ALLOCS_PURGED	The total number of allocate requests purged because the queue time exceeds the MAXQTIME value. Reset characteristic: reset to zero
Number of transactions attached	ISR_TRANS_ATTACHED	The total number of transactions attached for this connection. Reset characteristic: reset to zero
Remote Terminal Starts	ISR_REMOTE_TERM_STARTS	The total number of **START** requests sent from a remote terminal. Reset characteristic: reset to zero
Transaction Routing requests	ISR_TR_REQUESTS	The number of transaction routing requests on this connection. Reset characteristic: reset to zero
Bytes Sent by Transaction Routing requests	ISR_TR_BYTES_SENT	The number of bytes sent on transaction routing requests. Reset characteristic: reset to zero
Bytes Rcvd by Transaction Routing requests	ISR_TR_BYTES_RECEIVED	The number of bytes received by transaction routing requests. Reset characteristic: reset to zero

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Send Sessions	ISR_SEND_SESSIONS	The defined number of send sessions. The actual number of sessions used depends also on the number of receive sessions defined in the partner system. When the connection is established, these values are exchanged and the lower value is used. Reset characteristic: not reset
Current Send Sessions	ISR_CURRENT_SEND_SESSIONS	The current number of send sessions in use. Reset characteristic: reset to current value
Peak Send Sessions	ISR_PEAK_SEND_SESSIONS	The peak number of send sessions in use. Reset characteristic: reset to current value
Allocates Failed - Other	ISR_ALLOCATES_FAILED_OTHER	The number of allocate requests that failed because of other reasons. Reset characteristic: reset to zero
Number of QUEUELIMIT allocates rejected	ISR_QLIMIT_ALLOC_REJECTS	The total number of allocate requests rejected because the QUEUELIMIT value is reached. Reset characteristic: reset to zero
Number of XISQUE allocate requests rejected	ISR_XISQUE_ALLOC_REJECTS	The total number of allocate requests rejected by an XISQUE global user exit program. Reset characteristic: reset to zero
Number of XISQUE allocate queue purges	ISR_XISQUE_ALLOC_QPURGES	The total number of allocate queue purges that have occurred because of an XISQUE request for this connection. Reset characteristic: reset to zero.

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Number of XISQUE allocates purged	ISR_XISQUE_ALLOCS_PURGED	The total number of allocate requests purged because XISQUE requests that allocate queues are purged (ISR_XISQUE_ALLOC_QPURGES) for this connection. If XISQUE does not subsequently cancel this instruction, any subsequent allocate requests are purged and included in this statistic, because the XISQUE purging mechanism is still in operation. Reset characteristic: reset to zero
Function Shipped Program requests	ISR_FS_PG_REQUESTS	The number of program control LINK requests for function shipping on this connection. Reset characteristic: reset to zero
Bytes Sent by Program requests	ISR_FS_PG_BYTES_SENT	The number of bytes sent on LINK requests. Reset characteristic: reset to zero
Bytes Received by Program requests	ISR_FS_PG_BYTES_RECEIVED	The number of bytes received on LINK requests. Reset characteristic: reset to zero
Function Shipped Interval Control requests	ISR_FS_IC_REQUESTS	The number of interval control requests for function shipping on this connection. Reset characteristic: reset to zero
Bytes Sent by Interval Control requests	ISR_FS_IC_BYTES_SENT	The number of bytes sent on interval control requests. Reset characteristic: reset to zero
Bytes Rcvd by Interval Control Requests	ISR_FS_IC_BYTES_RECEIVED	The number of bytes received by interval control requests. Reset characteristic: reset to zero
Function Shipped File Control requests	ISR_FS_FC_REQUESTS	The number of file control requests for function shipping on this connection. Reset characteristic: reset to zero

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Bytes Sent by File Control Requests	ISR_FS_FC_BYTES_SENT	The number of bytes sent by file control requests. Reset characteristic: reset to zero
Bytes Rcvd by File Control Requests	ISR_FS_FC_BYTES_RECEIVED	The number of bytes received by file control requests. Reset characteristic: reset to zero
Function Shipped Transient Data Requests	ISR_FS_TD_REQUESTS	The number of transient data requests for function shipping on this connection. Reset characteristic: reset to zero
Bytes Sent by Transient Data Requests	ISR_FS_TD_BYTES_SENT	The number of bytes sent by transient data requests. Reset characteristic: reset to zero
Bytes Rcvd by Transient Data Requests	ISR_FS_TD_BYTES_RECEIVED	The number of bytes received by transient data requests. Reset characteristic: reset to zero
Function Shipped Temporary Storage Requests	ISR_FS_TS_REQUESTS	The number of temporary storage requests for function shipping on this connection. Reset characteristic: reset to zero
Bytes Sent by Temporary Storage Requests	ISR_FS_TS_BYTES_SENT	The number of bytes sent by temporary storage requests. Reset characteristic: reset to zero
Bytes Rcvd by Temporary Storage Requests	ISR_FS_TS_BYTES_RECEIVED	The number of bytes received by temporary storage requests. Reset characteristic: reset to zero
Unsupported Requests	ISR_UNSUPPORTED_REQUESTS	The number of attempts to route requests for unsupported function across this connection. Reset characteristic: reset to zero

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	ISR_IPCONN_GMT_CREATE_TIME	The date and time when the IPCONN was autoinstalled. The time shown is GMT. If the IPCONN was not autoinstalled, this field is not shown.
Not in DFHSTUP report	ISR_IPCONN_GMT_DELETE_TIME	The date and time when the autoinstalled IPCONN was deleted. The time shown is GMT. If the IPCONN was not autoinstalled, this field is not shown.
Not in DFHSTUP report	ISR_SSL_SUPPORT	Whether secure socket layer (SSL) authentication is supported. SSL_YES SSL_NO Reset characteristic: not reset
Not in DFHSTUP report	ISR_USERAUTH	The type of user authentication used. DEFAULTUSER IDENTIFY LOCAL VERIFY Reset characteristic: not reset
Not in DFHSTUP report	ISR_LINKAUTH	The type of link authentication used. CERTUSER SECUSER Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset

Table 112. IPCONN: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	ISR_IPCONN_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	ISR_IPCONN_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	ISR_MIRRORLIFE	The minimum lifetime of the mirror task for function-shipped requests received by this region. REQUEST TASK UOW Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related concepts:

"Interpreting IPCONN statistics" on page 561
Information about the purpose of IPCONN statistics.

Related reference:

"IPCONN report" on page 837
The IPCONN report shows information and statistics about IPCONN resource definitions, which define IP interconnectivity (IPIC) connections.

Related information:

EXTRACT STATISTICS in Reference > System programming

IPCONN: Summary resource statistics

A summary listing of resource statistics for each IPCONN. You can use IPCONN statistics to detect problems with IP interconnectivity (IPIC) connections.

Summary resource statistics are not available online.

IPIC is described in , in the *CICS Intercommunication Guide*.

Table 113. IPCONN: summary resource statistics

DFHSTUP name	Description
IPCONN Name	The name of an IPIC connection defined by an IPCONN definition in the CSD, or by autoinstall.
Autoinstalled IPCONN Create Date / Time	The date and time when the IPCONN was autoinstalled. The time shown is local time. If the IPCONN was not autoinstalled, this field is not shown.
Autoinstalled IPCONN Delete Date / Time	The date and time when the autoinstalled IPCONN was deleted. The time shown is local time. If the IPCONN was not autoinstalled, this field is not shown.
IPCONN Applid	The APPLID of the remote system, as specified in its system initialization table.
IPCONN Network ID	The network ID (that is, the z/OS Communications Server NETID or, for non-z/OS Communications Server systems, the value of the UOWNETQL system initialization parameter) of the remote system. This ID is used, in combination with the APPLID, to ensure unique naming for connecting systems. The name can be up to 8 characters in length and follows assembler language rules. It must start with an alphabetic character. This attribute is optional. If it is not specified, the z/OS Communications Server NETID (or, for non-z/OS Communications Server systems, the value of the UOWNETQL system initialization parameter) of the CICS on which the definition is installed is used.
TCPIPSERVICE name	The name of the PROTOCOL(IPIC) TCPIPSERVICE definition that defines the attributes of the inbound processing for this connection.
IPCONN Port Number	The decimal number of the port that is combined with the HOST value to specify the destination for outbound requests on this connection.
IPCONN Host	The host name of the target system for this connection.
IPCONN IP Family	The address format of the IP Resolved Address.
IPCONN IP Resolved Address	The IPv4 or IPv6 address of the host.
Receive Sessions	The defined number of receive sessions.
Peak Receive Sessions	The peak number of receive sessions in use for this connection.
Total Allocates	The total number of allocate requests for this connection.
Peak Allocates Queued	The peak number of allocate requests that have been queued for this connection.
Allocates Failed - Link	The number of allocate requests that failed because the connection is released or out-of-service.
Allocate queue limit	The value of the QUEUELIMIT parameter specified on the IPCONN definition. This value is the maximum number of allocate requests that CICS is to queue while waiting for free sessions.

Table 113. IPCONN: summary resource statistics (continued)

DFHSTUP name	Description
Maximum queue time (seconds)	The MAXQTIME specified on the IPCONN definition. This value represents the maximum time that queued allocate requests, waiting for free sessions on a connection that appears to be unresponsive, can wait. The maximum queue time is used only if a queue limit is specified for QUEUELIMIT; and the time limit is applied only when the queue length has reached the queue limit value.
Number of MAXQTIME allocate queue purges	The total number of times an allocate queue has been purged because of 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	The total number of allocate requests purged because the queue time exceeds the MAXQTIME value.
Number of transactions attached	The total number of transactions attached for this connection.
Function Shipped Program requests	The number of program control LINK requests for function shipping on this connection.
Bytes Sent by Program requests	The number of bytes sent on LINK requests.
Bytes Received by Program requests	The number of bytes received on LINK requests.
Function Shipped Interval Control requests	The number of interval control requests for function shipping on this connection.
Bytes Sent by Interval Control Requests	The number of bytes sent by interval control requests.
Bytes Rcvd by Interval Control Requests	The number of bytes received by interval control requests.
Send Sessions	The defined number of send sessions. The actual number of sessions used depends also on the number of receive sessions defined in the partner system. When the connection is established, these values are exchanged and the lower value is used.
Peak Send Sessions	The peak number of send sessions in use.
Allocates Failed - Other	The number of allocate requests that failed because of other reasons.
Number of QUEUELIMIT allocates rejected	The total number of allocate requests rejected because the QUEUELIMIT value is reached.
Number of XISQUE allocates rejected	The total number of allocate requests rejected by an XISQUE global user exit program.
Number of XISQUE allocate queue purges	The total number of allocate queue purges that have occurred because of an XISQUE request for this connection.
Number of XISQUE allocates purged	The total number of allocate requests purged because XISQUE requests that allocate queues are purged (ISR_XISQUE_ALLOC_QPURGES) for this connection. If XISQUE has not subsequently canceled this instruction, any subsequent allocate requests are purged and included in this statistic, because the XISQUE purging mechanism is still in operation.
Remote Terminal Starts	The total number of **START** requests sent from a remote terminal.
Transaction Routing requests	The number of transaction routing requests on this connection.

Table 113. IPCONN: summary resource statistics (continued)

DFHSTUP name	Description
Bytes Sent by Transaction Routing requests	The number of bytes sent on transaction routing requests.
Bytes Rcvd by Transaction Routing requests	The number of bytes received by transaction routing requests.
Function Shipped File Control requests	The number of file control requests for function shipping on this connection.
Bytes Sent by File Control Requests	The number of bytes sent by file control requests.
Bytes Rcvd by File Control Requests	The number of bytes received by file control requests.
Function Shipped Temporary Storage Requests	The number of temporary storage requests for function shipping on this connection.
Bytes Sent by Temporary Storage Requests	The number of bytes sent by temporary storage requests.
Bytes Rcvd by Temporary Storage Requests	The number of bytes received by temporary storage requests.
Function Shipped Transient Data Requests	The number of transient data requests for function shipping on this connection.
Bytes Sent by Transient Data Requests	The number of bytes sent by transient data requests.
Bytes Rcvd by Transient Data Requests	The number of bytes received by transient data requests.
Unsupported Requests	The number of attempts to route requests for unsupported function across this connection.

Journalname statistics

CICS collects statistics on the data written to each journal which can be used to analyze the activity of a single region.

Journalname statistics contain data about the use of each journal, as follows:
- The journal type (MVS logger, SMF, or dummy)
- The log stream name for MVS logger journal types only
- The number of API journal writes
- The number of bytes written
- The number of flushes of journal data to log streams or SMF.

Note that the CICS system journalname statistics for the last three items on this list are always zero.

Journalnames are a convenient means of identifying a destination log stream that is to be written to. CICS applications write data to journals with journalname. CICS itself usually uses the underlying log stream name when issuing requests to the CICS log manager, and this must be considered when interpreting journalname and log stream resource statistics. For example, these may show many operations against a log stream, but relatively few, if any, writes to a journalname which maps to that log stream. This indicates that it is CICS that accesses the resource at the log stream level, not an application writing to it through the CICS application programming interface. These results can typically be seen when examining the journalname resource statistics for DFHLOG and DFHSHUNT, and comparing them with the resource statistics for their associated CICS system log streams.

For more information about logging and journaling, see Logging and journaling performance.

Related reference:

"Journalnames report" on page 841
The Journalnames report is produced using a combination of the **EXEC CICS INQUIRE JOURNALNAME** and **EXEC CICS EXTRACT STATISTICS JOURNALNAME** commands. The statistics data is mapped by the **DFHLGRDS DSECT**.

Journalname: Resource statistics

You can retrieve Journalname statistics by using the **EXEC CICS EXTRACT STATISTICS JOURNALNAME** system command. They are mapped by the DFHLGRDS DSECT.

For more information about logging and journaling, see Chapter 16, "CICS logging and journaling: Performance and tuning," on page 225. For the system logs DFHLOG and DFHSHUNT, CICS does not use the journal for writing purposes, but writes directly to the log stream. So for these journals, "N/A" appears in the report under the headings Write requests, Bytes written and Buffer flushes.

These statistics fields contain the resource data that is collected by the log manager domain.

Table 114. Journalname: Resource statistics

DFHSTUP name	Field name	Description
Journal Name	LGRJNLNAME	The journal name. Reset characteristic: not reset
Journal Type	LGRJTYPE	The type of journal: MVS, SMF, or dummy. Reset characteristic: not reset
Log Stream Name	LGRSTREAM	The log stream name that is associated with the journal. Only journals that are defined as type MVS have associated log streams. The same log stream can be associated with more than one journal. Reset characteristic: not reset
Write Requests	LGRWRITES	The total number of times that a journal record was written to the journal. Reset characteristic: reset to zero

Table 114. Journalname: Resource statistics (continued)

DFHSTUP name	Field name	Description
Bytes Written	LGRBYTES	The total number of bytes written to the journal. Reset characteristic: reset to zero
Buffer Flushes	LGRBUFLSH	The total number of times that a journal block was written to the log stream (in the case of a journal that is defined as type MVS), or to the System Management Facility (in the case of a journal that is defined as type SMF). Journal blocks are flushed in the following circumstances: • An application executes an **EXEC CICS WRITE JOURNALNAME** or **EXEC CICS WRITE JOURNALNUM** command with the WAIT option. • An application executes an **EXEC CICS WAIT JOURNALNAME** or **EXEC CICS WAIT JOURNALNUM** command. • The journal buffer is full. This applies only to journals defined as type SMF (journals that are defined as type MVS use log stream buffers). • The log stream buffer is full. This applies only to journals defined as type MVS. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Journalname: Summary resource statistics

Journalname summary resource statistics are not available online.

These statistics fields contain the journalname summary resource data. For the system logs DFHLOG and DFHSHUNT, CICS does not use the journal for writing purposes, but writes directly to the log stream. So for these journals, “N/A” appears in the summary report under the headings Write requests, Bytes written and Buffer flushes.

Table 115. Journalname: Summary resource statistics

DFHSTUP name	Description
Journal Name	is the journal name.
Journal Type	is the journal type: • MVS • SMF • dummy
Log Stream Name	is the name of the log stream associated with the journal.
Write Requests	is the total number of times that a journal record was written to the journal.

Table 115. Journalname: Summary resource statistics (continued)

DFHSTUP name	Description
Bytes Written	is the total number of bytes written.
Buffer Flushes	is the total number of times that a journal block was written to the log stream (in the case of a journal defined as type MVS), or to the System Management Facility (in the case of a journal defined as type SMF).

JVM server statistics

CICS collects statistics for JVM servers and for Java programs that run in JVMs. You can use these statistics to manage and tune the Java workloads that are running in your CICS regions.

You can gather the following statistics related to Java:

- JVM server statistics, which tell you about the activity of the JVM that is used by a particular JVM server.
- JVM program statistics, which tell you about Java programs that run in JVM servers.

For information about how to tune JVM servers, see Improving Java performance in Improving performance.

Related reference:

"JVM Programs report" on page 842
The JVM Programs report shows information and statistics about Java programs that run in JVM servers or pooled JVMs. This report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** commands. The statistics data is mapped by the **DFHPGRDS DSECT**.

Related information:

Improving Java performance in Improving performance

JVMSERVER statistics

The JVM (SJ) domain collects statistics for JVM servers, including statistics on heap storage and garbage collection. Each JVM server is represented by a JVMSERVER resource.

You can get some information about the JVM server by inquiring on the JVMSERVER resource. The resource provides information such as the initial, maximum, and current heap size and the garbage collection policy that is being used by Java. Unlike pooled JVMs, the garbage collection is handled by Java automatically depending on the policy that is specified.

The DFH0STAT and DFHSTUP statistics programs provide more in-depth information about a JVM server:

- The statistics report how long Java applications are waiting for threads in the JVM server. If the waits are high and many tasks are suspended with the JVMTHRD wait, you can increase the value of the THREADLIMIT attribute on the JVMSERVER resource to make more threads available to the applications.

- The statistics report the heap sizes of the JVM. If the heap size after garbage collection is close to the maximum heap size, garbage collection might be occurring too often and you might need to increase the maximum heap size. If the peak heap size is much lower than the maximum heap size, you can either run more work in the JVM server, or edit the JVM profile and reduce the maximum heap size to save on storage.
- The statistics report the system threads in the JVM server. System threads are used to collect statistics and are also used by inquire and browse commands, but not by applications. You can find out how many times the JVM server was accessed for information and the associated processor usage. If the number is high, you might change the statistics interval or stop the inquire and browse requests.
- The statistics report major and minor garbage collection events. Minor garbage collection is only available on certain policies, so you might want to change the policy based on the information in the statistics.

These statistics can be a good starting point for tuning the performance of your Java workload.

JVMSERVER: Resource statistics

You can retrieve JVMSERVER statistics by using the **EXEC CICS EXTRACT STATISTICS JVMSERVER** system command. They are mapped by the DFHSJSDS DSECT.

Table 116. JVMSERVER: resource statistics

DFHSTUP name	Field name	Description
JVMSERVER name	SJS_JVMSERVER_NAME	The name of the JVMSERVER resource. Reset characteristic: not reset
JVMSERVER profile name	SJS_JVMSERVER_JVMPROFILE	The name of the JVM profile that is specified on the JVMSERVER resource. Reset characteristic: not reset
JVMSERVER LE runtime options	SJS_JVMSERVER_LE_RUNOPTS	The name of the Language Environment runtime options program that is specified on the JVMSERVER resource. Reset characteristic: not reset
JVMSERVER use count	SJS_JVMSERVER_USE_COUNT	The number of times the JVM server has been called. Reset characteristic: reset to zero
JVMSERVER thread limit	SJS_JVMSERVER_THREAD_LIMIT	The maximum number of threads in the JVM server. Reset characteristic: not reset
JVMSERVER current threads	SJS_JVMSERVER_THREAD_CURRENT	The current number of threads in the JVM server. Reset characteristic: not reset

Table 116. JVMSERVER: resource statistics (continued)

DFHSTUP name	Field name	Description
JVMSERVER peak threads	SJS_JVMSERVER_THREAD_HWM	The peak number of threads in the JVM server. Reset characteristic: reset to current value (SJS_JVMSERVER_THREAD_CURRENT)
JVMSERVER thread limit waits	SJS_JVMSERVER_THREAD_WAITS	The number of tasks that waited for a free thread. Reset characteristic: reset to zero
JVMSERVER thread limit wait time	SJS_JVMSERVER_THREAD_WAIT_TIME	The amount of time in seconds that tasks waited for a free thread. Reset characteristic: reset to zero
JVMSERVER current thread waits	SJS_JVMSERVER_THREAD_WAIT_CUR	The number of tasks that are currently waiting for a free thread. Reset characteristic: reset to zero
JVMSERVER peak thread waits	SJS_JVMSERVER_THREAD_WAIT_HWM	The peak number of tasks that waited for a free thread. Reset characteristic: reset to number of tasks current waiting (SYS_JVMSERVER_THREAD_WAIT_CURR)
JVMSERVER system thread use count	SJS_JVMSERVER_SYS_USE_COUNT	The number of times that the system thread has been used. Reset characteristic: reset to zero
JVMSERVER system thread waits	SJS_JVMSERVER_SYS_WAITED	The number of CICS tasks that waited for a system thread. Reset characteristic: reset to zero
JVMSERVER system thread wait time	SJS_JVMSERVER_SYS_WAITED_TIME	The accumulated time in seconds that tasks spent waiting for a system thread. Reset characteristic: reset to zero
JVMSERVER current sys thread waits	SJS_JVMSERVER_SYS_WAIT_CUR	The current number of tasks that are waiting for a system thread. Reset characteristic: not reset

Table 116. JVMSERVER: resource statistics (continued)

DFHSTUP name	Field name	Description
JVMSERVER peak system thread waits	SJS_JVMSERVER_SYS_WAIT_HWM	The highest number of tasks that waited for a system thread. Reset characteristic: reset to current number of waiting tasks (SJS_JVMSERVER_SYS_WAIT_CURR)
JVMSERVER creation time of JVM	SJS_JVMSERVER_JVM_CREATION_LCL	The time stamp (STCK) in local time of when the JVM was created for the JVM server. Reset characteristic: not reset
JVMSERVER status	SJS_JVMSERVER_STATE	The state of the JVMSERVER resource. Reset characteristic: not reset
JVMSERVER current heap size	SJS_JVMSERVER_CURRENT_HEAP	The size in bytes of the heap that is currently allocated to the JVM server. Reset characteristic: not reset
JVMSERVER initial heap size	SJS_JVMSERVER_INITIAL_HEAP	The size in bytes of the initial heap that is allocated to the JVM server. This value is set by the **-Xms** option in the JVM profile. Reset characteristic: not reset
JVMSERVER maximum heap size	SJS_JVMSERVER_MAX_HEAP	The size in bytes of the maximum heap that can be allocated to the JVM server. This value is set by the **-Xmx** option in the JVM profile. Reset characteristic: not reset
JVMSERVER peak heap size	SJS_JVMSERVER_PEAK_HEAP	The size in bytes of the largest heap that has been allocated to the JVM server. Reset characteristic: not reset
JVMSERVER heap occupancy	SJS_JVMSERVER_OCCUPANCY	The size in bytes of the heap immediately after the last garbage collection occurred. Reset characteristic: not reset
JVMSERVER Garbage Collection (GC)	SJS_JVMSERVER_GC_POLICY	The garbage collection policy that is being used by the JVM. Reset characteristic: not reset

Table 116. JVMSERVER: resource statistics (continued)

DFHSTUP name	Field name	Description
JVMSERVER no. of major GC events	SJS_JVMSERVER_MJR_GC_EVENTS	The number of major garbage collection events that have occurred. Reset characteristic: reset to zero
JVMSERVER total elapsed time spent in major GC	SJS_JVMSERVER_MJR_GC_CPU	The total elapsed time in milliseconds that was spent performing major garbage collection. Reset characteristic: reset to zero
JVMSERVER total memory freed by major GC	SJS_JVMSERVER_MJR_HEAP_FREED	The total memory in bytes that was freed by performing major garbage collection. Reset characteristic: reset to zero
JVMSERVER no. of minor GC events	SJS_JVMSERVER_MNR_GC_EVENTS	The number of minor garbage collections that have occurred. Reset characteristic: reset to zero
JVMSERVER total elapsed time spent in minor GC	SJS_JVMSERVER_MNR_GC_CPU	The total elapsed time in milliseconds that was spent performing minor garbage collection. Reset characteristic: reset to zero
JVMSERVER total memory freed by minor GC	SJS_JVMSERVER_MNR_HEAP_FREED	The total memory in bytes that was freed by performing minor garbage collection. Reset characteristic: reset to zero
Not in DFHSTUP report	SJS_JVMSERVER_JVM_CREATION_GMT	The time stamp (STCK) in GMT of when the JVM was created for the JVM server. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset

Table 116. JVMSERVER: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	SJS_JVMSERVER_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	SJS_JVMSERVER_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

JVMSERVER: Summary resource statistics

A summary listing of resource statistics for JVM servers, including the number of times the JVM server has been used by Java applications and thread usage.

Summary statistics are not available online.

Table 117. JVMSERVER: Summary resource statistics

DFHSTUP name	Description
JVMSERVER name	The name of the JVMSERVER resource.
JVMSERVER LE runtime options	The name of the program that defines the runtime options of the Language Environment enclave.
JVMSERVER use count	The number of times that the JVM server has been called.
Thread limit	The maximum number of threads that are allowed to run in the JVM server.

Table 117. JVMSERVER: Summary resource statistics (continued)

DFHSTUP name	Description
Peak threads	The peak number of threads in the JVM server.
Thread limit waits	The number of tasks that waited for a free thread.
Thread limit wait time	The amount of time that tasks waited for a free thread.
Peak thread limit waits	The peak number of tasks that waited for a free thread.
System thread use count	The number of times that the system thread has been used.
System thread waits	The number of CICS tasks that waited for a system thread.
System thread wait time	The accumulated time that tasks spent waiting for a system thread.
Current sys thread waits	The current number of tasks that are waiting for a system thread.
Peak system thread waits	The highest number of tasks that waited for a system thread.
JVMSERVER status	The status of the JVMSERVER resource.
Current heap size	The size in bytes of the heap that is currently allocated to the JVM server.
Initial heap size	The size in bytes of the initial heap that is allocated to the JVM server. This value is set by the **-Xms** option in the JVM profile.
Max heap size	The size in bytes of the maximum heap that can be allocated to the JVM server. This value is set by the **-Xmx** option in the JVM profile.
Peak heap size	The size in bytes of the largest heap that has been allocated to the JVM server.
Heap occupancy	The size in bytes of the heap immediately after the last garbage collection occurred.
Garbage Collection (GC)	The garbage collection policy that is being used by the JVM.
Number of major GC events	The number of major garbage collection events that have occurred.
Elapsed time in major GC	The elapsed time that was spent performing major garbage collection.
Total memory freed by major GC	The total memory that was freed by performing major garbage collection.
Number of minor GC events	The number of minor garbage collections that have occurred.
Elapsed time in minor GC	The elapsed time that was spent performing minor garbage collection.
Total memory freed by minor GC	The total memory that was freed by performing minor garbage collection.

JVM program statistics

JVM program statistics are collected for every installed JVM program in the CICS region that runs in a JVM server. Statistics for programs that run in a JVM are collected separately from statistics for other programs, because the Java programs are not loaded by CICS.

For public JVM programs, these statistics are mapped by the DFHPGRDS DSECT. For private JVM programs for applications that are deployed on platforms, these statistics are mapped by the DFHPGPDS DSECT. The statistics records for private JVM programs have information about the application for which the JVM program was defined.

JVM programs that are defined as application entry points are identified by a field in the statistics record. When interval statistics, end-of-day statistics, requested statistics, requested reset statistics, or unsolicited statistics are produced for a JVM program that is defined as an application entry point, two statistics records are written, one mapped by the DFHPGRDS DSECT for public JVM programs, and one mapped by the DFHPGPDS DSECT for private JVM programs.

Viewing statistics for Java programs

CICS does not collect statistics for Java programs when an **EXEC CICS EXTRACT STATISTICS PROGRAM** command is issued. To see them, you must use the **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** command instead. However, when you browse program names by using the **EXEC CICS INQUIRE PROGRAM** command, Java programs are found. An application that collects statistics for programs by browsing with the **EXEC CICS INQUIRE PROGRAM** command, and then issuing the **EXEC CICS EXTRACT STATISTICS PROGRAM** command for the program names that it finds, would receive a “not found” response when it attempted to collect statistics for any Java programs.

To avoid receiving this response, make the application check the RUNTIME value for each program name that it finds. If the RUNTIME value is JVM, the application must not issue the **EXEC CICS EXTRACT STATISTICS PROGRAM** command for that program name. If you want to see the statistics for programs with a RUNTIME value of JVM, you can make the application issue the **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** command for those programs. The statistics information that is collected for Java programs is not the same as the statistics information collected for other programs.

Java programs that run in a JVM have their own DFH0STAT report, the JVM Programs report. The DFH0STAT report for Program Totals also includes a figure for the number of Java programs, but this figure is obtained using the JVMPROGRAM keyword.

JVM programs - Public: Resource statistics

You can retrieve statistics for public JVM programs by using the **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** system command. They are mapped by the DFHPGRDS DSECT

JVM program resource statistics for public JVM programs show information and statistics about each public JVM program, including the JVM profile that is used.

Statistics for public JVM programs are mapped by the DFHPGRDS DSECT. For private JVM programs for applications that are deployed on platforms, these statistics are mapped by the DFHPGPDS DSECT, which has information about the application for which the JVM program was defined. For details of the DSECT and DFHSTUP report for private JVM programs, see “JVM programs - Private: Resource statistics” on page 585.

JVM programs that are defined as application entry points are identified by the PGR_JVMPROGRAM_ENTRYPOINT field. Both public and private statistics records are written for these programs, mapped once by each DSECT.

Table 118. JVM programs - Public: Resource statistics

DFHSTUP name	Field name	Description
Program name	PGR_JVMPROGRAM_NAME	The name of the Java program. Reset characteristic: not reset
Not in DFHSTUP report	PGR_JVMPROGRAM_ENTRYPOINT	Whether the program is defined as an application entry point for an application deployed on a platform. Reset characteristic: not reset
Times used	PGR_JVMPROGRAM_USECOUNT	The number of times the program has been used. Reset characteristic: reset to zero
Exec key	PGR_JVMPROGRAM_EXEC_KEY	The execution key that the program requires, either CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource. Programs that run in a JVM server always run in CICS key. Reset characteristic: not reset
JVM class	PGR_JVMPROGRAM_JVMCLASS	The main class in the program as specified in the JVMCLASS attribute of the PROGRAM resource. Reset characteristic: not reset
JVM server	PGR_JVMPROGRAM_SERVER	The name of the JVMSERVER resource that the program requires to run in a JVM server, as specified in the JVMSERVER attribute of the PROGRAM resource. Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

JVM programs - Public: Summary resource statistics

A summary listing of resource statistics for all public Java programs that run in a JVM.

Summary statistics are not available online.

Private JVM programs for applications that are deployed on platforms are reported in a separate summary report. For details of this report, see "JVM programs - Private: Summary resource statistics" on page 587. JVM programs that are defined as application entry points appear in both the public and private resource summary reports.

Table 119. JVM programs - Public: Summary resource statistics

DFHSTUP name	Description
Program name	The name of the Java program.
JVM server	The name of the JVMSERVER resource that the program requires to run in a JVM server, as specified in the JVMSERVER attribute of the PROGRAM resource.
Times used	The number of times the program has been used.
Exec key	Java programs that run in a JVM server always use CICS key.
JVM class	The main class in the program, as specified in the JVMCLASS attribute of the PROGRAM resource.

JVM programs - Private: Resource statistics

You can retrieve statistics for private Java programs by using the **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** system command. They are mapped by the DFHPGPDS DSECT.

JVM program resource statistics for private JVM programs show information and statistics about the private JVM programs for applications that are deployed on platforms.

Statistics for private Java programs for applications that are deployed on platforms are mapped by the DFHPGPDS DSECT. For public Java programs, these statistics are mapped by the DFHPGRDS DSECT. For details of the DSECT and DFHSTUP report for public JVM programs, see "JVM programs - Public: Resource statistics" on page 583.

JVM programs that are defined as application entry points are identified by an application operation being named in the PGP_JVMPROGRAM_OPERATION_NAME field. Both public and private statistics records are written for these programs, mapped once by each DSECT.

The DFHSTUP report shows the private Java programs for each application that is deployed on a platform. For programs that are declared as application entry points, the report shows the application operation that is named for the application entry point.

Table 120. JVM programs - Private: Resource statistics

DFHSTUP name	Field name	Description
Platform	PGP_JVMPROGRAM_PLATFORM_NAME	The name of the platform where the application that uses the private programs is deployed. Reset characteristic: not reset
Application	PGP_JVMPROGRAM_APPLICATION_NAME	The name of the application that uses the private programs. Reset characteristic: not reset
Major version	PGP_JVMPROGRAM_APPL_MAJOR_VER	The major version number of the application that uses the private programs. Reset characteristic: not reset
Minor version	PGP_JVMPROGRAM_APPL_MINOR_VER	The minor version number of the application that uses the private programs. Reset characteristic: not reset
Micro version	PGP_JVMPROGRAM_APPL_MICRO_VER	The micro version number of the application that uses the private programs. Reset characteristic: not reset
Program name	PGP_JVMPROGRAM_NAME	The name of the Java program. Reset characteristic: not reset
Times used	PGP_JVMPROGRAM_USECOUNT	The number of times the program has been used. Reset characteristic: reset to zero
Exec key	PGP_JVMPROGRAM_EXEC_KEY	The execution key that the program requires, either CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource. Programs that run in a JVM server always run in CICS key. Reset characteristic: not reset

Table 120. JVM programs - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
JVM class	PGP_JVMPROGRAM_JVMCLASS	The main class in the program as specified in the JVMCLASS attribute of the PROGRAM resource. Reset characteristic: not reset
JVM server	PGP_JVMPROGRAM_SERVER	The name of the JVMSERVER resource that the program requires to run in a JVM server, as specified in the JVMSERVER attribute of the PROGRAM resource. Reset characteristic: not reset
Operation	PGP_JVMPROGRAM_OPERATION_NAME	For programs that are declared as application entry points, the application operation that is named for the application entry point. Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

JVM programs - Private: Summary resource statistics

A summary listing of resource statistics for all private Java programs for applications that are deployed on platforms.

Summary statistics are not available online.

Public JVM programs are reported in a separate summary report. For details of this report, see "JVM programs - Public: Summary resource statistics" on page 584. JVM programs that are defined as application entry points appear in both the public and private resource summary reports.

Table 121. JVM programs - Private: Summary resource statistics

DFHSTUP name	Description
Platform	The name of the platform where the application that uses the private programs is deployed.
Application	The name of the application that uses the private programs.
Major version	The major version number of the application that uses the private programs.
Minor version	The minor version number of the application that uses the private programs.
Micro version	The micro version number of the application that uses the private programs.
Program name	The name of the Java program.
Operation	For programs that are declared as application entry points, the application operation that is named for the application entry point.
JVM server	The name of the JVMSERVER resource that the program requires to run in a JVM server, as specified in the JVMSERVER attribute of the PROGRAM resource.
Times used	The number of times the program has been used.
Exec key	Java programs that run in a JVM server always use CICS key.

Table 121. JVM programs - Private: Summary resource statistics (continued)

DFHSTUP name	Description
JVM class	The main class in the program, as specified in the JVMCLASS attribute of the PROGRAM resource.

LIBRARY statistics

LIBRARY statistics report resource data for dynamic program LIBRARY concatenations, which are data sets from which program load modules can be loaded.

For public LIBRARY resources, these statistics are mapped by the DFHLDBDS DSECT. For private LIBRARY resources for applications that are deployed on platforms, these statistics are mapped by the DFHLDYDS DSECT. The statistics records for private LIBRARY resources have information about the application for which the LIBRARY resource was defined.

LIBRARY - Public: Resource statistics

You can retrieve statistics for public LIBRARY resources by using the **EXEC CICS EXTRACT STATISTICS LIBRARY** system command. They are mapped by the DFHLDBDS DSECT.

These statistics fields contain the resource data collected by the loader for each dynamic program LIBRARY concatenation.

Statistics for public LIBRARY resources are mapped by the DFHLDBDS DSECT. For private LIBRARY resources for applications that are deployed on platforms, these statistics are mapped by the DFHLDYDS DSECT, which has information about the application for which the LIBRARY resource was loaded. For details of the DSECT and DFHSTUP report for private LIBRARY resources, see “LIBRARY - Private: Resource statistics” on page 593.

Table 122. LIBRARY - Public: Resource statistics

DFHSTUP name	Field name	Description
LIBRARY name	LDB_LIBRARY_NAME	The name of the library. Reset characteristic: not reset

Table 122. LIBRARY - Public: Resource statistics (continued)

DFHSTUP name	Field name	Description
Search position	LDB_LIBRARY_SEARCH_POS	The current absolute position of this library in the overall library search order. The first enabled library in the search order will have a search position of 1, the next library will have a search position of 2, and so on. The search position is not the same as the ranking, although its value is determined by the relative ranking values of the various library resources in the system. The search position values, relative to other library resources with the same ranking value, are indeterminate, but their search position values relative to each other are retained across a warm or emergency restart. The relative search position values of library resources with the same ranking are not guaranteed to be the same after a cold or initial start. If the library is disabled, the search position is 0, indicating that the library does not participate in the overall search. Reset characteristic: not reset
Ranking	LDB_LIBRARY_RANKING	Indicates where this library appears in the overall library search order, relative to other library concatenations. A lower number indicates that this library is searched for programs to load before other library resources with higher ranking numbers. Reset characteristic: not reset

Table 122. LIBRARY - Public: Resource statistics (continued)

DFHSTUP name	Field name	Description
Critical	LDB_LIBRARY_CRITICAL	Indicates whether the library is critical to CICS starting. The values are as follows: **Yes** The LIBRARY is critical to CICS starting. If the LIBRARY cannot be successfully installed during CICS startup for any reason, then a GO or CANCEL message is issued. The operator decides whether to override the critical status and allow CICS to start. If CICS is allowed to continue, the LIBRARY is installed in a DISABLED status, unless installation was not possible at all; for example, because of a short-on-storage condition. If the reply is to continue with the startup, the LIBRARY is not recataloged as NONCRITICAL, so the critical status is explicitly set to NONCRITICAL if it is decided that the LIBRARY is not to be regarded as critical in future. **No** The LIBRARY is not critical to CICS startup. If the LIBRARY cannot be successfully installed during CICS startup, the LIBRARY is left in an installed but disabled state and a warning message is issued, but CICS startup continues. Reset characteristic: not reset

Table 122. LIBRARY - Public: Resource statistics (continued)

DFHSTUP name	Field name	Description
Enable status	LDB_LIBRARY_ENABLE_STATUS	Identifies whether the LIBRARY is included in the overall LIBRARY search order. The values are as follows: **DISABLED** The LIBRARY is disabled, and is not currently included in the LIBRARY search order. The data sets in this LIBRARY concatenation are not searched for program artifacts to load. **DISABLING** A request to disable the LIBRARY was received, but is still being processed. **ENABLED** The LIBRARY is enabled, and is currently included in the LIBRARY search order. The data sets in this LIBRARY concatenation searched for program artifacts to load. **ENABLING** A request to enable the LIBRARY was received, but is still being processed. **DISCARDING** A request to discard the LIBRARY from the CICS system was received, but is still being processed. Reset characteristic: not reset
Program loads	LDB_LIBRARY_PROG_LOADS	The number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS-managed storage. Reset characteristic: reset to zero
Not in DFHSTUP report	LDB_LIBRARY_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	LDB_LIBRARY_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	LDB_LIBRARY_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset

Table 122. LIBRARY - Public: Resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	LDB_LIBRARY_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	LDB_LIBRARY_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	LDB_LIBRARY_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	LDB_LIBRARY_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
Number Dsnames	LDB_LIBRARY_NUMDSNAMES	The number of data sets in the LIBRARY concatenation. For a dynamically defined LIBRARY, this number indicates the non blank DSNAMExx values, and cannot be a value larger than 16. For the statically defined DFHRPL, this number indicates the data sets in the concatenation, and can be a value larger than 16. Reset characteristic: not reset
Concatenation	Not in the DSECT; added when report is formatted	The concatenation number of the data set in the LIBRARY concatenation. Reset characteristic: not reset
LIBRARY Dsname	LDB_DSNAME	The 44-character name of each data set in the LIBRARY concatenation. If this library is dynamically defined, these are the data sets specified on the LIBRARY definition, all but one of which can be blank. If this DFHRPL is the statically defined one, these are the first 16 data sets in the DFHRPL concatenation, or as many data sets as are specified up to 16, with the remaining DSNAMExx fields being blank. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource

signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

LIBRARY - Private: Resource statistics

You can retrieve statistics for private LIBRARY resources by using the **EXEC CICS EXTRACT STATISTICS LIBRARY** system command. They are mapped by the DFHLDYDS DSECT.

These statistics fields contain the resource data collected by the loader for each dynamic program LIBRARY concatenation for applications that are deployed on platforms.

Statistics for private LIBRARY resources for applications that are deployed on platforms are mapped by the DFHLDYDS DSECT. For public LIBRARY resources, these statistics are mapped by the DFHLDBDS DSECT. For details of the DSECT and DFHSTUP report for public LIBRARY resources, see “LIBRARY - Public: Resource statistics” on page 588.

Table 123. LIBRARY - Private: Resource statistics

DFHSTUP name	Field name	Description
Platform	LDY_LIBRARY_PLATFORM_NAME	The name of the platform where the application that uses the private LIBRARY resources is deployed. Reset characteristic: not reset
Application	LDY_LIBRARY_APPLICATION_NAME	The name of the application that uses the private LIBRARY resources. Reset characteristic: not reset
Major version	LDY_LIBRARY_APPL_MAJOR_VER	The major version number of the application that uses the private LIBRARY resources. Reset characteristic: not reset
Minor version	LDY_LIBRARY_APPL_MINOR_VER	The minor version number of the application that uses the private LIBRARY resources. Reset characteristic: not reset
Micro version	LDY_LIBRARY_APPL_MICRO_VER	The micro version number of the application that uses the private LIBRARY resources. Reset characteristic: not reset

Table 123. LIBRARY - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
LIBRARY name	LDY_LIBRARY_NAME	The name of the LIBRARY resource. Reset characteristic: not reset
Search position	LDY_LIBRARY_SEARCH_POS	The current absolute position of this library in the overall library search order. The first enabled library in the search order has a search position of 1, the next library has a search position of 2, and so on. The search position is not the same as the ranking, although its value is determined by the relative ranking values of the various library resources in the system. The search position values, relative to other library resources with the same ranking value, are indeterminate, but their search position values relative to each other are retained across a warm or emergency restart. The relative search position values of library resources with the same ranking are not guaranteed to be the same after a cold or initial start. If the library is disabled, the search position is 0, indicating that the library does not participate in the overall search. Reset characteristic: not reset
Ranking	LDY_LIBRARY_RANKING	Indicates where this library appears in the overall library search order, relative to other library concatenations. A lower number indicates that this library is searched for programs to load before other library resources with higher ranking numbers. Reset characteristic: not reset
Critical	LDY_LIBRARY_CRITICAL	This attribute does not apply to private LIBRARY resources for applications deployed on platforms. Reset characteristic: not reset

Table 123. LIBRARY - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
Enable status	LDY_LIBRARY_ENABLE_STATUS	Identifies whether the LIBRARY is included in the overall LIBRARY search order. The values are as follows: **DISABLED** The LIBRARY is disabled, and is not currently included in the LIBRARY search order. The data sets in this LIBRARY concatenation are not searched for program artifacts to load. **DISABLING** A request to disable the LIBRARY was received, but is still being processed. **ENABLED** The LIBRARY is enabled, and is currently included in the LIBRARY search order. The data sets in this LIBRARY concatenation searched for program artifacts to load. **ENABLING** A request to enable the LIBRARY was received, but is still being processed. **DISCARDING** A request to discard the LIBRARY from the CICS system was received, but is still being processed. Reset characteristic: not reset
Program loads	LDY_LIBRARY_PROG_LOADS	The number of times the loader has issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS-managed storage. Reset characteristic: reset to zero
Not in DFHSTUP report	LDY_LIBRARY_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	LDY_LIBRARY_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset

Table 123. LIBRARY - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	LDY_LIBRARY_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	LDY_LIBRARY_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	LDY_LIBRARY_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	LDY_LIBRARY_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	LDY_LIBRARY_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
Number Dsnames	LDY_LIBRARY_NUMBER_DSNAMES	The number of data sets in the LIBRARY concatenation. For a dynamically defined LIBRARY, this number indicates the non blank DSNAMExx values, and cannot be a value larger than 16. For the statically defined DFHRPL, this number indicates the data sets in the concatenation, and can be a value larger than 16. Reset characteristic: not reset
Concatenation	Not in the DSECT; added when report is formatted	The concatenation number of the data set in the LIBRARY concatenation. Reset characteristic: not reset

Table 123. LIBRARY - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
LIBRARY Dsname	LDY_LIBRARY_DSNAME	The 44-character name of each data set in the LIBRARY concatenation. If this library is dynamically defined, these are the data sets specified on the LIBRARY definition, all but one of which can be blank. If this DFHRPL is the statically defined one, these are the first 16 data sets in the DFHRPL concatenation, or as many data sets as are specified up to 16, with the remaining DSNAMExx fields being blank. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Loader domain statistics

Related concepts:

"Interpreting loader statistics"
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.

Related reference:

"Loader and Program Storage report" on page 847
The Loader and Program Storage report is produced using a combination of the **EXEC CICS EXTRACT STATISTICS PROGRAM** and **EXEC CICS EXTRACT STATISTICS STORAGE** commands. The statistics data is mapped by the **DFHLDGDS** and **DFHSMDDS DSECT**.

Interpreting loader statistics

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.

"Average loading time" = "Total loading time" / "Number of library load requests". This statistic indicates the response time of tasks when accessing a program that must be brought into storage. 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 released by using a freemain request, in the 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 statistic indicates 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 (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 value is meaningful only if there has been significant loader domain activity during the interval and might be distorted by startup usage patterns.

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

This statistic indicates the response time impact that a task might suffer because of contention for loader domain resources.

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

Loader domain: Global statistics

These statistics fields contain the global data collected by the loader domain. The loader domain maintains global statistics to assist the user in tuning and accounting.

These statistics can be retrieved using the **EXTRACT STATISTICS PROGRAM** system command, and are mapped by the DFHLDGDS DSECT.

Table 124. Loader domain: Global statistics — All Areas

DFHSTUP name	Field name	Description
Library load requests	LDGLLR	The number of times the loader issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Modules in the LPA are not included in this value. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. Reset characteristic: reset to zero
Library load requests on the RO TCB	LDGLLRRO	The number of times the loader issued a program load request that used the RO (resource-owning) TCB. This value is a subset of the number of library loads shown by LDGLLR. To calculate the number of program load requests that ran on open TCBs, subtract this value from the value shown by LDGLLR. Reset characteristic: reset to zero

Table 124. Loader domain: Global statistics — All Areas (continued)

DFHSTUP name	Field name	Description
Total loading time	LDGLLT	The time taken for the number of library loads shown by LDGLLR. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. 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
Total loading time on the RO TCB	LDGLLTRO	The time taken for the number of library loads shown by LDGLLRRO. This value is a subset of the time shown by LDGLLT. To calculate the time taken for program load requests that ran on open TCBs, subtract this value from the value shown by LDGLLT. 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
Average loading time		The average time taken to load a program. The value is an average including both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. 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
Average loading time on the RO TCB		The average time taken to complete only those program load requests that used the RO (resource-owning) TCB. 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	The number of uses of any program by the CICS system. Reset characteristic: not reset

Table 124. Loader domain: Global statistics — All Areas (continued)

DFHSTUP name	Field name	Description
Waiting requests	LDGWLR	The number of loader domain requests that are currently waiting for the loader domain to complete an operation on the program on behalf of another task. Program load requests might wait for the following reasons: • The program is being loaded by another task that is running on an open TCB. • The loader domain is searching the link pack area (LPA) for the program. • A NEWCOPY request or physical load is in progress for the program. Reset characteristic: not reset
Requests that waited	LDGWTDLR	The number of loader domain requests that waited for the loader domain to complete an operation on the program on behalf of another task. This figure is the number of tasks that waited in the past, and does not include tasks that are currently waiting (LDGWLR). Program load requests might wait for the following reasons: • The program is being loaded by another task that is running on an open TCB. • The loader domain is searching the link pack area (LPA) for the program. • A NEWCOPY request or physical load is in progress for the program. Reset characteristic: reset to zero
Peak waiting Loader requests	LDGWLRHW	The maximum number of tasks suspended at one time. Reset characteristic: reset to current value (LDGWLR)
Times at peak	LDGHWMT	The number of times the high-water mark 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	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

Table 124. Loader domain: Global statistics — All Areas (continued)

DFHSTUP name	Field name	Description
Times DFHRPL re-opened	LDGDREBS	The number of times the loader received an end-of-extent condition during a load operation, and successfully closed and re-opened the DFHRPL or dynamic LIBRARY concatenation and retried the load operation. Reset characteristic: reset to zero

Loader domain: Global statistics — CDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset

Loader domain: Global statistics — CDSA

DFHSTUP name	Field name	Description
Amount of DSA occupied by Not In Use programs	LDGSTGNIU	The current amount of CDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: Global statistics — ECDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset

Loader domain: Global statistics — ECDSA

DFHSTUP name	Field name	Description
Amount of DSA occupied by Not In Use programs	LDGSTGNIU	The current amount of ECDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: Global statistics — SDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset

Loader domain: Global statistics — SDSA

DFHSTUP name	Field name	Description
Amount of DSA occupied by Not In Use programs	LDGSTGNIU	The current amount of SDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: Global statistics — ESDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset

Loader domain: Global statistics — ESDSA

DFHSTUP name	Field name	Description
Amount of DSA occupied by Not In Use programs	LDGSTGNIU	The current amount of ESDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Loader domain: Global statistics — RSDA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset

Loader domain: Global statistics — ERDSA

DFHSTUP name	Field name	Description
Programs removed by compression	LDGDPSCR	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	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 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		The average length of time that a program is eligible for removal from storage by the DPSC mechanism. This value is calculated by DFHSTUP. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*. Reset characteristic: none
Reclaims from Not In Use queue	LDGRECNIU	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	LDGPROGNIU	The number of programs on the Not-In-Use (NIU) queue. Reset characteristic: not reset
Amount of DSA occupied by Not In Use programs	LDGSTGNIU	The current amount of ERDSA storage which is occupied by Not-In-Use (NIU) programs. Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

Loader domain: Summary global statistics

These statistics fields contain the summary global data for the loader.

Summary statistics are not available online.

Table 125. Loader domain: Summary global statistics

DFHSTUP name	Description
Library load requests	The number of times the loader issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Modules in the LPA are not included in this value. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB.
Library load requests on the RO TCB	The number of times the loader issued a program load request that used the RO (resource-owning) TCB. This value is a subset of the number of library loads shown by “Library load requests”. To calculate the number of program load requests that ran on open TCBs, subtract this value from the value shown by “Library load requests”.
Total loading time	The time taken for the number of library loads shown by “Library load requests”. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Total loading time on the RO TCB	The time taken for the number of library loads shown by “Library load requests on the RO TCB”. This value is a subset of the time shown by “Total loading time”. To calculate the time taken for program load requests that ran on open TCBs, subtract this value from the value shown by “Total loading time”. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Average loading time	The average time taken to load a program. The value is an average including both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. DFHSTUP expresses this time as *hours:minutes:seconds.decimals*.
Average loading time on the RO TCB	The average time taken to complete only those program load requests that used the RO (resource-owning) TCB. DFHSTUP expresses this time as *hours:minutes:seconds.decimals*.
Program uses	The number of uses of any program by the CICS system.
Requests that waited	The number of loader domain requests that waited for the loader domain to complete an operation on the program on behalf of another task. Program load requests might wait for the following reasons: • The program is being loaded by another task that is running on an open TCB. • The loader domain is searching the link pack area (LPA) for the program. • A NEWCOPY request or physical load is in progress for the program.
Peak waiting Loader requests	The maximum number of tasks suspended at one time.

Table 125. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Times at peak	The number of times the high-water mark indicated by LDGWLRHW was reached. This, along with the previous 2 values, is an indication of the level of contention for loader resource.
Total waiting time	The 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	The number of times the loader received an end-of-extent condition during a load operation, and successfully closed and re-opened the DFHRPL or dynamic LIBRARY concatenation and retried the load operation.
CDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.
ECDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.

Table 125. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.
SDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.

Table 125. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
ESDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.
RDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.

Table 125. Loader domain: Summary global statistics (continued)

DFHSTUP name	Description
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.
ERDSA	
Programs removed by compression	The total number of program instances removed from storage by the Dynamic Program Storage Compression (DPSC) mechanism.
Total Not In Use queue membership time	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 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	The average time between a program becoming eligible for removal from storage by the DPSC and the actual time of its removal from storage. The DFHSTUP report expresses this time as *hours:minutes:seconds.decimals*.
Reclaims from Not In Use queue	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	The total number of programs on the Not-In-Use (NIU) queue.

Logstream statistics

CICS collects statistics on the data written to each log stream which can be used to analyze the activity of a single region. However, because log streams can be shared across multiple MVS images, it can be more useful to examine the statistics generated by MVS.

Log stream statistics contain data about the use of each log stream including the following:

- The number of write requests to the log stream
- The number of bytes written to the log stream
- The number of log stream buffer waits
- The number of log stream browse and delete requests.

The CICS system log stream statistics for the last three items on this list are always zero.

Journalnames are a convenient means of identifying a destination log stream that is to be written to. CICS applications write data to journals using their journalname. CICS itself usually uses the underlying log stream name when issuing requests to the CICS log manager, and this must be considered when interpreting journalname and log stream resource statistics. For example, the statistics might show many operations against a log stream, but relatively few, if any, writes to a journalname which maps to that log stream. This indicates that it is CICS that accesses the resource at the log stream level, not an application writing to it through the CICS application programming interface. The results can typically be seen when examining the journalname resource statistics for DFHLOG and DFHSHUNT, and comparing them with the resource statistics for their associated CICS system log streams.

For more information about logging and journaling, see Logging and journaling performance.

Related reference:

"Logstreams reports" on page 851
Four Logstream reports are produced using the **EXEC CICS EXTRACT STATISTICS** STREAMNAME and **EXEC CICS INQUIRE** STREAMNAME commands. The statistics data is mapped by the **DFHLGGDS DSECT**.

Logstream: Global statistics

You can retrieve global log stream statistics by using the **EXEC CICS EXTRACT STATISTICS STREAMNAME** system command. They are mapped by the DFHLGGDS DSECT.

These statistics fields contain the global data collected by the log manager domain.

For more information about logging and journaling, see Chapter 16, "CICS logging and journaling: Performance and tuning," on page 225.

Table 126. Logstream: Global statistics

DFHSTUP name	Field name	Description
Activity Keypoint Frequency (AKPFREQ)	LGGAKPFREQ	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. This is the AKPFREQ value specified in the SIT, or as an override, or changed dynamically. Reset characteristic: not reset
Activity Keypoints Taken	LGGAKPSTKN	The number of activity keypoints taken. Reset characteristic: reset to zero
Log Deferred Force (LGDFINT) Interval (msec)	LGGLGDEFER	The current log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. This is the LGDFINT value specified in the SIT, or as an override, or changed dynamically. Reset characteristic: not reset

Related information:

EXTRACT STATISTICS in Reference > System programming

Logstream: Resource statistics

You can retrieved log stream resource statistics by using the **EXEC CICS EXTRACT STATISTICS STREAMNAME** system command. They are mapped by the DFHLGSDS DSECT.

These statistics fields contain the resource data collected by the log manager domain.

For more information about logging and journaling, see Chapter 16, "CICS logging and journaling: Performance and tuning," on page 225.

Table 127. Logstream: Resource statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	The logstream name. Reset characteristic: not reset
System Log	LGSSYSLG	Indicates if the logstream forms part of the System Log. Reset characteristic: not reset
Structure Name	LGSSTRUC	The coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams. Reset characteristic: not reset
Max Block Length	LGSMAXBL	The maximum block size allowed by the MVS Logger for the logstream. Reset characteristic: not reset
DASD Only	LGSDONLY	Indicates the type of logstream. If set to 'YES' the logstream is of type DASDONLY. If set to 'NO' the logstream is of type coupling facility (CF). Reset characteristic: not reset
Retention Period	LGSRETPD	The logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Reset characteristic: not reset

Table 127. Logstream: Resource statistics (continued)

DFHSTUP name	Field name	Description
Auto Delete	LGSAUTOD	The log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Reset characteristic: not reset
Delete Requests	LGSDELETES	The number of DELETEs of blocks of data from the logstream. For non-system logs, the report will show 'N/A' here, as CICS does not issue Log Delete requests against non-system logs. Reset characteristic: reset to zero
Query Requests	LGSQUERIES	The number of queries that CICS made to check the status of the logstream. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Logstream: Request statistics

You can retrieve log stream request statistics by using the **EXEC CICS EXTRACT STATISTICS STREAMNAME** system command. They are mapped by the DFHLGSDS DSECT.

These statistics fields contain the request data collected by the log manager domain.

Table 128. Logstream: Request statistics

DFHSTUP name	Field name	Description
Log Stream Name	LGSTRNAM	is the logstream name. Reset characteristic: not reset
Write Requests	LGSWRITES	is the number of WRITEs of blocks of data to the logstream. Reset characteristic: reset to zero
Bytes Written	LGSBYTES	is the total number of bytes written to the logstream Reset characteristic: reset to zero
Buffer Appends	LGSBUFAPP	is the number of occasions on which a journal record was successfully appended to the current logstream buffer. Reset characteristic: reset to zero

Table 128. Logstream: Request statistics (continued)

DFHSTUP name	Field name	Description
Waits Buff Full	LGSBUFWAIT	is the total number of attempts made to append a journal record to the current logstream buffer while the buffers were logically full. This situation arises when the current logstream buffer has insufficient space to accommodate the journal record, and I/O is already in progress for the alternate logstream buffer. Reset characteristic: reset to zero
Current Frce Wtrs	LGSCUFWTRS	is the current number of tasks suspended while requesting a flush of the logstream buffer currently in use. Reset characteristic: not reset
Peak Frce Wtrs	LGSPKFWTRS	is the peak number of tasks suspended while requesting a flush of the logstream buffer currently in use. Reset characteristic: reset to current
Total Force Wts	LGSTFCWAIT	is the total number of tasks suspending while requesting a flush of the logstream buffer currently in use. Reset characteristic: reset to zero
Browse Starts	LGSBRWSTRT	is the number of BROWSE operations started on the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these. Reset characteristic: reset to zero
Browse Reads	LGSBRWREAD	is the number of READs of blocks of data from the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these. Reset characteristic: reset to zero
Retry Errors	LGSRTYERRS	is the number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the logstream. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Logstream: Summary global statistics

These statistics fields contain the logstream summary global data.

Summary statistics are not available online.

Table 129. Logstream: Summary global statistics

DFHSTUP name	Description
Activity Keypoint Frequency (AKPFREQ)	The last activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. This is the last AKPFREQ value as specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM AKP(fullword binary data-value)** command.
Total Activity Keypoints Taken	The total number of activity keypoints taken.
Log Deferred Force (LGDFINT) Interval (msec)	The last log deferral interval, which is the period of time used by CICS Log Manager when determining how long to delay a forced journal write request before invoking the MVS system logger. This is the last LGDFINT value that was specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM LOGDEFER(halfword binary data-value)** command.

Logstream: Summary resource statistics

These statistics fields contain the logstream summary resource data.

Summary statistics are not available online.

Table 130. Logstream: Summary resource statistics

DFHSTUP name	Description
Log Stream Name	is the logstream name.
System Log	indicates if the logstream forms part of the System Log.
Structure Name	is the coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams.
Max Block Length	is the maximum block size allowed by the MVS Logger for the logstream.
DASD Only	indicates the type of logstream. If set to 'YES' the logstream is of type DASDONLY. If set to 'NO' the logstream is of type coupling facility (CF).
Retention Period	is the logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger.
Auto Delete	is the log data auto delete indicator. If set to 'YES' the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to 'NO' the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period.
Log Delete Requests	is the total number of DELETEs of blocks of data from the logstream. For non-system logs, the report will show 'N/A' here, as CICS does not issue Log Delete requests against non-system logs.

Table 130. Logstream: Summary resource statistics (continued)

DFHSTUP name	Description
Log Query Requests	is the total number of queries that CICS made to check the status of the logstream.

Logstream: Summary request statistics

These statistics fields contain the logstream summary request data.

Summary statistics are not available online.

Table 131. Logstream: Summary request statistics

DFHSTUP name	Description
Log Stream Name	is the logstream name.
Write Requests	is the total number of WRITEs of blocks of data to the logstream.
Bytes Written	is the total number of bytes written to the logstream.
Buffer Appends	is the total number of occasions on which a journal record was successfully appended to the current logstream buffer.
Waits Buffer Full	is the total number of attempts made to append a journal record to the current logstream while the buffers were logically full.
Peak Force Wtrs	is the peak number of tasks suspended while requesting a FLUSH of the logstream buffer currently in use.
Total Force Waits	is the total number of tasks suspended while requesting a FLUSH of the logstream buffer currently in use.
Log Browse Starts	is the total number of BROWSE operations started on the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these.
Log Browse Reads	is the total number of READs of blocks of data from the logstream. For non-system log logstreams, the report will show 'N/A' here, as you cannot browse these.
Retry Errors	is the total number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the logstream.

LSR pool statistics

CICS supports the use of up to 255 LSR pools, and produces two sets of statistics for LSR pool activity.

Related concepts:

"Interpreting LSR pool statistics"
CICS supports the use of up to 255 LSR pools. CICS produces two sets of statistics for LSR pool activity: one set detailing the activity for each LSR pool, and one set giving details for each file associated with an LSR pool. Statistics are printed for all pools that have been built (a pool is built when at least one file that uses the pool has been opened).

Related reference:

"LSR pools report" on page 854
The LSR pools report is produced using the **EXEC CICS EXTRACT STATISTICS LSRPOOL** command. The statistics data is mapped by the **DFHA08DS DSECT**.

Interpreting LSR pool statistics

CICS supports the use of up to 255 LSR pools. CICS produces two sets of statistics for LSR pool activity: one set detailing the activity for each LSR pool, and one set giving details for each file associated with an LSR pool. Statistics are printed for all pools that have been built (a pool is built when at least one file that uses the pool has been opened).

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

When the last open file in an LSR pool is closed, the pool is deleted. The subsequent unsolicited statistics (USS) LSR pool 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) can be those that 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 was built.

It is possible to change the LSR pool 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 might fail. If the pool is not built and the pool characteristics are specified by you, ensure that these are adequate for the file. If the pool is not built and CICS calculates all or some of the operands, it can 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 processor savings of keeping data in memory, using the relatively cheap expanded storage, while allowing central storage to be used more effectively.

LSR pool: Resource statistics for each LSR pool

You can retrieve LSR pool resource statistics by using the **EXEC CICS EXTRACT STATISTICS LSRPOOL** system command. They are mapped by the DFHA08DS DSECT.

The following information describes the size and characteristics of the pool, and shows the data collected for the use of strings and buffers.

Table 132. LSR pool: Resource statistics for each LSR pool

DFHSTUP name	Field name	Description
Pool Number	A08SRPID	The identifying number of the pool. This value must be in the range 1 through 255. Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	A08FLAGS	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	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	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	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

Table 132. LSR pool: Resource statistics for each LSR pool (continued)

DFHSTUP name	Field name	Description
NOT IN DFHSTUP REPORT	A08GBKDD	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	The length of the largest key of a VSAM data set that can use the LSR pool. The value is obtained from one of the following sources: • 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	The value obtained from one of the following sources: • 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	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	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

Table 132. LSR pool: Resource statistics for each LSR pool (continued)

DFHSTUP name	Field name	Description
Peak concurrently active strings	A08BKHAS	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 higher 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.

Related information:

EXTRACT STATISTICS in Reference > System programming

LSR pool: Data buffer statistics

Table 133. LSR pool: Data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	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	The number of data buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_DATA	The number of successful lookasides to data buffers for the pool. Reset characteristic: not reset
Reads	A08TOFRD_DATA	The number of read I/Os to the data buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_DATA	The number of user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset

Table 133. LSR pool: Data buffer statistics (continued)

DFHSTUP name	Field name	Description
Non-user writes	A08TONUW_DATA	The number of non-user-initiated buffer WRITEs from data buffers for the pool. Reset characteristic: not reset

LSR pool: Hiperspace data buffer statistics

Table 134. LSR pool: Hiperspace data buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	The size of the buffers that are available to CICS. Buffers can 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	The number of Hiperspace data buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_DATA	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	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	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	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

LSR pool: Index buffer statistics

Table 135. LSR pool: Index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	The size of the buffers that are available to CICS. Buffers can 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	The number of index buffers used by the pool. Reset characteristic: not reset
Lookasides	A08TOBFF_INDX	The number of successful lookasides to index buffers for the pool. Reset characteristic: not reset
Reads	A08TOFRD_INDX	The number of read I/Os to the index buffers for the pool. Reset characteristic: not reset
User writes	A08TOUIW_INDX	The number of user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset
Non-user writes	A08TONUW_INDX	The number of non-user-initiated buffer WRITEs from index buffers for the pool. Reset characteristic: not reset

LSR pool: Hiperspace index buffer statistics

The following group of statistics fields describes the characteristics and usage of the different buffer sizes available for use by the pool.

LSR pool Hiperspace index buffer 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.

Table 136. LSR pool: Hiperspace index buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	The size of the buffers that are available to CICS. Buffers can 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	The number of Hiperspace index buffers specified for the pool Reset characteristic: not reset
Hiperspace reads	A08TOCRS_INDX	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	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	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	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. Reset characteristic: not reset

LSR pool: Buffer statistics

Table 137. LSR pool: Buffer statistics

DFHSTUP name	Field name	Description
Buffer Size	A08BKBSZ	The size of the buffers that are available to CICS. Buffers can 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

Table 137. LSR pool: Buffer statistics (continued)

DFHSTUP name	Field name	Description
Number	A08BKBFN	The number of buffers of each size available to CICS: Reset characteristic: not reset
Lookasides	A08BKBFF	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 must 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	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	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
Non-user writes	A08BKNUW	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

LSR pool: Hiperspace buffer statistics

Table 138. LSR pool: Hiperspace buffer statistics

DFHSTUP name	Field name	Description
Size	A08BKBSZ	The size of the buffers that are available to CICS. Buffers can 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	The number of Hiperspace buffers specified for the pool. Reset characteristic: not reset
Hiperspace reads	A08BKCRS	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers. Reset characteristic: not reset
Hiperspace writes	A08BKCWS	The number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Reset characteristic: not reset
Hiperspace failed reads	A08BKCRF	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	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.

LSR pool: Summary resource statistics for each LSR pool

Summary statistics are unavailable online.

Table 139. LSR pool: Summary resource statistics for each LSR pool

DFHSTUP name	Description
Total number of pools built	The total number of LSR pools that were built during the entire CICS run.

Table 139. LSR pool: Summary resource statistics for each LSR pool (continued)

DFHSTUP name	Description
Peak requests that waited for string	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	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.
Peak concurrently active strings	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 higher than this value in the statistics, you could consider reducing it so that your pool of VSAM strings is not bigger than you need.

LSR pool: Summary data buffer statistics

The following group of statistics fields summarizes the usage of each of the 255 LSR pools during the entire CICS run.

Summary statistics are unavailable online.

Table 140. LSR pool: Summary data buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.
Lookasides	The total number of successful lookasides to data buffers for the pool.
Reads	The total number of read I/O operations to the data buffers for the pool.
User writes	The total number of user-initiated buffer WRITE requests from data buffers for the pool.
Non-user writes	The total number of non-user-initiated buffer WRITE requests from data buffers for the pool.

LSR pool: Summary Hiperspace data buffer statistics

Summary statistics are unavailable online.

Table 141. LSR pool: Summary Hiperspace data buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.

Table 141. LSR pool: Summary Hiperspace data buffer statistics (continued)

DFHSTUP name	Description
Hiperspace reads	The total number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers.
Hiperspace writes	The total number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers.
Hiperspace failed reads	The total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	The total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

LSR pool: Summary index buffer statistics

Summary statistics are unavailable online.

Table 142. LSR pool: Summary index buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.
Lookasides	The total number of successful lookasides to index buffers for the pool.
Reads	The total number of read I/O operations to the index buffers for the pool.
User writes	The total number of user-initiated buffer WRITE requests from index buffers for the pool.
Non-user writes	The total number of non-user-initiated buffer WRITE requests from index buffers for the pool.

LSR pool: Summary Hiperspace index buffer statistics

Summary statistics are unavailable online.

Table 143. LSR pool: Summary Hiperspace index buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.
Hiperspace reads	The total number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers.

Table 143. LSR pool: Summary Hiperspace index buffer statistics (continued)

DFHSTUP name	Description
Hiperspace writes	The total number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers.
Hiperspace failed reads	The total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	The total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD.

LSR pool: Summary buffer statistics

Summary statistics are unavailable online.

Table 144. LSR pool: Summary buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.
Lookasides	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 employed involves either increasing the number of buffers of a particular CI size until the ratio of lookasides to READ requests stops increasing significantly or, conversely, reducing the number of buffers until the ratio of lookasides to READ requests 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	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	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.

Table 144. LSR pool: Summary buffer statistics (continued)

DFHSTUP name	Description
Non-user writes	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.

LSR pool: Summary Hiperspace buffer statistics

Summary statistics are unavailable online.

Table 145. LSR pool: Summary Hiperspace buffer statistics

DFHSTUP name	Description
Pool Number	The identifying number of the pool. This value must be in the range 1 through 255.
Hiperspace reads	The total number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers.
Hiperspace writes	The total number of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers.
Hiperspace failed reads	The total number of CREAD requests that failed. MVS had withdrawn the space and VSAM had to read data from DASD.
Hiperspace failed writes	The total number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write data to DASD. The 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.

If the allocation of files to the LSR pool is changed during the period that the statistics cover, no history of this is available and only the current list of files sharing the pool are printed in this section. The activity of all files that have used the pool are, however, included in all the preceding sections of these statistics.

LSR pool: Files - Resource statistics for each file specified to use the pool

Table 146. LSR pool: Files - Resource statistics for each file specified to use the pool

DFHSTUP name	Field name	Description
Pool Number	A09SRPID	The LSR pool number, in the range 1 through 255, associated with this file. Reset characteristic: not reset

Table 146. LSR pool: Files - Resource statistics for each file specified to use the pool (continued)

DFHSTUP name	Field name	Description
File Name	A09DSID	The CICS file identifier you specified through resource definition online. Reset characteristic: not reset
Data Buff Size	A09DBN	The buffer size used for the file's data records. This value is one of the 11 possible VSAM buffer sizes ranging from 512-bytes to 32 KB. The value is zero if the file has not been opened yet. Reset characteristic: not reset
Index Buff Size	A09IBN	The buffer size used for the file's index records. This is printed, even if the file has later been dynamically allocated to a VSAM RRDS. The values this field can take are the same as for the data buffer size statistic. Reset characteristic: not reset
Total Buff Waits	A09TBW	The number of requests that must 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	The peak number of requests that must 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

LSR pool: Files - Summary resource statistics

Summary statistics are unavailable online.

Table 147. LSR pool: Files - Summary resource statistics

DFHSTUP name	Description
Pool Number	The LSR pool number, in the range 1 through 255, associated with this file.
File Name	The CICS file identifier you specified through resource definition online.

Table 147. LSR pool: Files - Summary resource statistics (continued)

DFHSTUP name	Description
Data Buff Size	The last non-zero value encountered for the buffer size used for the file's data records. This value is one of the 11 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.
Index Buff Size	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 later been dynamically allocated to a VSAM RRDS. This field can take are the same values as the data buffer size statistic.
Total Buff Waits	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.
Peak Buff Waits	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 domain statistics

You can use monitoring domain statistics to measure the amount of CPU, storage, temporary-storage requests, and other resources used, by task. This information provides a view of the performance of the CICS system.

Related reference:

"System Status report" on page 885
The System Status report is produced from various sources. The commands used are detailed in the table.

Monitoring domain: global statistics

You can retrieve monitoring domain global statistics by using the **EXEC CICS EXTRACT STATISTICS MONITOR** system command. They are mapped by the DFHMNGDS DSECT.

These statistics fields are collected from the monitoring domain.

Table 148. Monitoring domain: global statistics

DFHSTUP name	Field name	Description
CEC Machine Type and Model Number	MNGMCHTP, MNGMDLID	The CEC machine type and model number for the physical hardware environment where the CICS region is running. CEC (central electronics complex) is a commonly used synonym for CPC (central processing complex). Reset characteristic: not reset

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Exception records	MNGER	The number of exception records written to SMF. Reset characteristic: reset to zero
Exception records suppressed	MNGERS	The number of exception records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
Performance records	MNGPR	The number of performance records scheduled for output to SMF. The monitoring domain buffers performance class records. If monitoring is deactivated, the performance class records that have been buffered are not in the report. Reset characteristic: reset to zero
Performance records suppressed	MNGPRS	The number of performance records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
Resource records	MNGRR	The number of transaction resource records scheduled for output to SMF. The monitoring domain buffers transaction resource class records. If monitoring is deactivated, the resource class records that have been buffered are not in the report. Reset characteristic: reset to zero
Resource records suppressed	MNGRRS	The number of resource records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
Identity records	MNGIR	The number of identity records scheduled for output to SMF. The monitoring domain buffers identity class records. If monitoring is deactivated, the identity class records that have been buffered are not in the report. Reset characteristic: reset to zero

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Identity records suppressed	MNGIRS	The number of identity records suppressed by the global user exit (XMNOUT). Reset characteristic: reset to zero
SMF records	MNGSMFR	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 each SMF record has one exception record. The performance class, for example, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class is deactivated, or CICS is quiescing. Reset characteristic: reset to zero
SMF errors	MNGSMFE	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
SMF Records Compressed	MNGSMFCM	The number of compressed monitoring records written to the SMF data set. This information is collected only when data compression for monitoring records is active. Reset characteristic: not reset
SMF Records Not Compressed	MNGSMFNC	The number of monitoring records written to the SMF data set for which data compression was not performed. This information is collected only when data compression for monitoring records is active. Reset characteristic: not reset
Average Compressed Record Length	MNGAVCRL	The rolling average compressed record length for monitoring records written to the SMF data set, calculated from those monitoring records that were compressed. This information is collected only when data compression for monitoring records is active. Reset characteristic: not reset

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Average Uncompressed Record Length	MNGAVURL	The rolling average record length for monitoring records written to the SMF data set for which data compression was not performed. This information is only collected when data compression for monitoring records is active. Reset characteristic: not reset
Data Compression Option	MNGMRCMP	Whether data compression is active for the CICS SMF 110 monitoring records produced by the CICS monitoring facility. Values are as follows: **0** Not active **1** Active Reset characteristic: not reset
DPL Resource Limit	MNGDPLRL	The maximum number of distributed program links for which transaction resource monitoring is being performed. Reset characteristic: not reset
File Resource Limit	MNGFRL	The maximum number of files for which transaction resource monitoring is being performed. Reset characteristic: not reset
Tsqueue Resource Limit	MNGTRL	The maximum number of temporary storage queues for which transaction resource monitoring is being performed. Reset characteristic: not reset
MVS WLM Mode	MNGWLMMD	The z/OS Workload Manager (WLM) mode that is in operation in the CICS region. Reset characteristic: not reset
MVS WLM Server	MNGWLMST	Whether the CICS region is a z/OS Workload Manager server. Reset characteristic: not reset

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
MVS WLM Service Class	MNGWLMSC	The class name of the z/OS Workload Manager service for the CICS region. Reset characteristic: not reset
MVS WLM Workload Name	MNGWLMWN	The name of the workload defined for the CICS region. Reset characteristic: not reset
MVS WLM Resource Group	MNGWLMRG	The name of the z/OS Workload Manager resource group, if any. Reset characteristic: not reset
MVS WLM Report Class	MNGWLMRC	The name of the z/OS Workload Manager report class, if any. Reset characteristic: not reset
MVS WLM Goal Type	MNGWLMGT	The z/OS Workload Manager goal type for the CICS address space, if any. Values are as follows: **0** Not applicable **1** Velocity **2** Discretionary **3** System Reset characteristic: not reset
MVS WLM CPU Critical	MNGWLMCC	Whether long-term processor protection is assigned to the CICS address space in the z/OS Workload Manager. Values are as follows: **0** Not critical **1** Critical Reset characteristic: not reset
MVS WLM Storage Critical	MNGWLMSK	Whether long-term storage protection is assigned to the CICS address space in the z/OS Workload Manager. Values are as follows: **0** Not critical **1** Critical Reset characteristic: not reset

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
WLM Address Space Goal Mgmt	MNGWLMGM	Whether z/OS Workload Manager manages the CICS address space using region goals, transaction goals, or both. Values are as follows: **0** Transaction goals **1** Region goals **2** Both goals Reset characteristic: not reset
MVS WLM Goal Value	MNGWLMGV	For a z/OS Workload Manager goal type of velocity, the goal value for the CICS address space, 1 - 99. For other goal types, this field is zero. Reset characteristic: not reset
MVS WLM Goal Importance	MNGWLMGI	The importance level of the z/OS Workload Manager goal for the CICS address space. Reset characteristic: not reset
--User transactions ended	MNGUTNUM	is the number of user transactions that have ended. Reset characteristic: reset to zero
System transactions ended	MNGSTNUM	is the number of system transactions that have ended. Reset characteristic: reset to zero
Time last user transaction attached	MNGLUTAT	is the date and time of the last transaction attach processed by the monitoring domain. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds:decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been ended since the statistics were last reset. Reset characteristic: reset to zero

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Time last user transaction ended	MNGLUTCL	is the date and time at which the last transaction ended. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds:decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been ended since the statistics were last reset. Reset characteristic: reset to zero
MXT at last user transaction attach	MNGMXUTA	The current MXT value at the time of the last transaction attached. Reset characteristic: reset to zero
Current tasks at last attach	MNGCAUTA	The current number of user transactions attached in the region at the time of the last transaction attached. Reset characteristic: reset to zero
Average user transaction resp time	MNGAUTRT	The rolling average user transaction response time. Reset characteristic: reset to zero
Peak user transaction resp time	MNGPUTRT	is the maximum user transaction response time (expressed in STCK units). Reset characteristic: reset to zero
Peak user transaction resp time at	MNGLUTRT	is the timestamp (expressed in STCK units in local time) of the maximum user transaction response time. Reset characteristic: reset to zero
Total transaction CPU time	MNGCPUT	The total transaction CPU time accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Reset characteristic: reset to zero

Table 148. Monitoring domain: global statistics (continued)

DFHSTUP name	Field name	Description
Total transaction CPU time on CP	MNGTONCP	The total transaction CPU time on a standard processor accumulated by the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Reset characteristic: reset to zero
Total transaction CPU offload on CP	MNGOFLCP	The total transaction CPU time on a standard processor but was eligible for offload to a specialty processor (zIIP or zAAP) accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Monitoring domain: summary global statistics

Monitoring domain summary global statistics are not available online.

Table 149. Monitoring domain: summary global statistics

DFHSTUP name	Description
CEC Machine Type and Model Number	The CEC machine type and model number for the physical hardware environment where the CICS region is running. CEC (central electronics complex) is a commonly used synonym for CPC (central processing complex).
Exception Records	The total number of exception records written to SMF.
Exception Records Suppressed	The total number of exception records suppressed by the global user exit (XMNOUT).
Performance Records	The total number of performance records scheduled for output to SMF. The monitoring domain buffers performance class records. If monitoring is deactivated, the performance class records that have been buffered are not in the report.
Performance Records Suppressed	The total number of performance records suppressed by the global user exit (XMNOUT).
Resource Class Records	The number of transaction resource records scheduled for output to SMF. The monitoring domain buffers transaction resource class records. If monitoring is deactivated, the resource class records that have been buffered are not in the report.

Table 149. Monitoring domain: summary global statistics (continued)

DFHSTUP name	Description
Resource Records Suppressed	The total number of resource records suppressed by the global user exit (XMNOUT).
Identity records	The total number of identity class records scheduled for output to SMF. The monitoring domain buffers identity class records. If monitoring is deactivated, the identity class records that have been buffered are not in the report.
Identity records suppressed	The total number of identity class records suppressed by the global user exit (XMNOUT).
SMF Records	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 each SMF record has one exception 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 is deactivated, or CICS is quiescing.
SMF Errors	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.
SMF Records Compressed	The number of compressed monitoring records written to the SMF data set. This information is collected only when data compression for monitoring records is active.
SMF Records Not Compressed	The number of monitoring records written to the SMF data set for which data compression was not performed. This information is collected only when data compression for monitoring records is active.
Average Compressed Record Length	The rolling average compressed record length for monitoring records written to the SMF data set, calculated from those monitoring records that were compressed. This information is collected only when data compression for monitoring records is active.
Average Uncompressed Record Length	The rolling average record length for monitoring records written to the SMF data set for which data compression was not performed. This information is only collected when data compression for monitoring records is active.
Data Compression Option	Whether data compression is active for the CICS SMF 110 monitoring records produced by the CICS monitoring facility. Values are as follows: **0** Not active **1** Active

Table 149. Monitoring domain: summary global statistics (continued)

DFHSTUP name	Description
File Resource Limit	The maximum number of files for which transaction resource monitoring is being performed.
Tsqueue Resource Limit	The maximum number of temporary storage queues for which transaction resource monitoring is being performed.
MVS WLM Mode	The z/OS Workload Manager (WLM) mode that is in operation in the CICS region.
MVS WLM Server	Whether the CICS region is a z/OS Workload Manager server.
MVS WLM Service Class	The class name of the z/OS Workload Manager service for the CICS region..
MVS WLM Workload Name	The name of the workload defined for the CICS region.
MVS WLM Resource Group	The name of the z/OS Workload Manager resource group, if any.
MVS WLM Report Class	The name of the z/OS Workload Manager report class, if any.
MVS WLM Goal Type	The z/OS Workload Manager goal type for the CICS address space, if any. Values are as follows: **0** Not applicable **1** Velocity **2** Discretionary **3** System
MVS WLM CPU Critical	Whether long-term processor protection is assigned to the CICS address space in the z/OS Workload Manager. Values are as follows: **0** Not critical **1** Critical
MVS WLM Storage Critical	Whether long-term storage protection is assigned to the CICS address space in the z/OS Workload Manager. Values are as follows: **0** Not critical **1** Critical
WLM Address Space Goal Mgmt	Whether z/OS Workload Manager manages the CICS address space using region goals, transaction goals, or both. Values are as follows: **0** Transaction goals **1** Region goals **2** Both goals

Table 149. Monitoring domain: summary global statistics (continued)

DFHSTUP name	Description
MVS WLM Goal Value	For a z/OS Workload Manager goal type of velocity, the goal value for the CICS address space, 1 - 99. For other goal types, this field is zero.
MVS WLM Goal Importance	The importance level of the z/OS Workload Manager goal for the CICS address space.
User transactions ended	The total number of user transactions that have ended.
System transactions ended	The total number of system transactions that have ended.
Total transaction CPU time	The total transaction CPU time accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed.
Total transaction CPU time on CP	The total transaction CPU time on a standard processor accumulated by the CICS dispatcher managed TCB modes used by the transactions that have completed.
Total transaction CPU offload on CP	The total transaction CPU time on a standard processor but was eligible for offload to a specialty processor (zIIP or zAAP) accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed.

Named counter sequence number server

Named counter sequence number server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

Named counter sequence number server statistics

The statistics are described in detail in the DFHNCS4D data area.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The individual fields of the structure have the following meanings.

Table 150. Named counter server: list structure statistics

Statistic name	Field	Description
Lists		
	S4NAME	Full name of list structure
	S4PREF	First part of structure name
	S4POOL	Pool name part of structure name
	S4CNNAME	Name for connection to structure
	S4CNPREF	Prefix for connection name
	S4CNSYSN	Own MVS system name from CVTSNAME
Size	S4SIZE	Current allocated size for the list structure.

Table 150. Named counter server: list structure statistics (continued)

Statistic name	Field	Description
Max size	S4SIZEMX	Maximum size to which this structure could be altered.
Entries		
In Use	S4ENTRCT	Number of entries currently in use.
Max Used	S4ENTRHI	Maximum number of entries in use (since last reset).
Min Free	S4ENTRLO	Minimum number of free entries (since last reset).
Total	S4ENTRMX	Total entries in the currently allocated structure (initially set at structure connection time and updated on completion of any structure alter request).
Requests		
Create	S4CRECT	Create counter
Get	S4GETCT	Get and increment counter
Set	S4SETCT	Set counter
Delete	S4DELCT	Delete counter
Inquire	S4KEQCT	Inquire KEQ
Browse	S4KGECT	Inquire KGE
Responses		
Asynch	S4ASYCT	Number of requests for which completion was asynchronous.
Unavail	S4RSP9CT	Structure temporarily unavailable, for example during rebuild.
Normal	S4RSP1CT	Number of normal responses.
Not Fnd	S4RSP2CT	The specified entry (table or item) was not found.
Vers Chk	S4RSP3CT	A version check failed for an entry being updated, indicating that another task had updated it first.
List Chk	S4RSP4CT	A list authority comparison failed, usually meaning that the table is in the process of being deleted.
Str Full	S4RSP5CT	The list structure became full.
I/O Err	S4RSP6CT	Some other error code was returned by IXLLIST.

Named counter server: storage statistics

These are statistics returned by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW. Storage in these pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage.

Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map.

If there are no free areas of the right size and there is not enough storage left in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The statistics are described in detail in the DFHNCS5D data area.

Table 151. Temporary storage data sharing: LOC=ANY usage statistics

Statistic name	Field	Description
Name	S5ANYNAM	Pool name AXMPGANY.
Size	S5ANYSIZ	Size of the storage pool area.
	S5ANYPTR	Address of storage pool area.
	S5ANYMX	Total pages in the storage pool.
In Use	S5ANYUS	Number of used pages in the pool.
Free	S5ANYFR	Number of free pages in the pool.
Min Free	S5ANYLO	The lowest free pages (since reset).
Gets	S5ANYRQG	Storage GET requests.
Frees	S5ANYRQF	Storage FREE requests.
Fails	S5ANYRQS	GETs which failed to obtain storage.
Retries	S5ANYRQC	Compress (defragmentation) attempts.

Table 152. Temporary storage data sharing: LOC=BELOW usage statistics

Statistic name	Field	Description
Name	S5LOWNAM	Pool name AXMPGLOW.
Size	S5LOWSIZ	Size of the storage pool area.
	S5LOWPTR	Address of the storage pool area.
	S5LOWMX	Total pages in the storage pool.
In Use	S5LOWUS	Number of used pages in the storage pool.
Free	S5LOWFR	Number of free pages in the storage pool.
Min Free	S5LOWLO	The lowest number of free pages (since reset).
Gets	S5LOWRQG	Storage GET requests.
Frees	S5LOWRQF	Storage FREE requests.
Fails	S5LOWRQS	GETs which failed to obtain storage.
Retries	S5LOWRQC	Compress (defragmentation) attempts.

Program autoinstall statistics

Related reference:

"Program Autoinstall report" on page 862
The Program Autoinstall report shows information and statistics about the status of program autoinstall, catalog program definitions, and the number of autoinstalls that were attempted, rejected, and failed.

Program autoinstall: Global statistics

You can retrieved program autoinstall global statistics by using the **EXEC CICS EXTRACT STATISTICS PROGAUTO** system command They are mapped by the DFHPGGDS DSECT.

Table 153. Program autoinstall: Global statistics

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 user-replaceable 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 user-replaceable program did not provide valid attributes; the model name specified by the user-replaceable program was not defined; the exit tried to recurse, and disabled the user-replaceable program. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Program autoinstall: Summary global statistics

Program autoinstall: Summary global statistics are not available online.

Table 154. Program autoinstall: Summary global statistics

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.

PIPELINE definition statistics

PIPELINE resource definitions are used in web services support when a CICS application is in the role of a web service provider or requester. They provide information about the message handler programs that act on a service request and on the response.

Statistics are provided for each PIPELINE resource definition, and a total use count for all PIPELINE definitions is also available. For information about the PIPELINE reports, see “PIPELINEs report” on page 859.

Related reference:

"PIPELINEs report" on page 859
The PIPELINEs report is produced using a combination of **EXEC CICS INQUIRE PIPELINE** and **EXEC CICS EXTRACT STATISTICS PIPELINE RESID()** commands. The statistics data is mapped by the **DFHPIRDS DSECT**.

PIPELINE definitions: Resource statistics

You can retrieve PIPELINE definition resource statistics by using the **EXEC CICS EXTRACT STATISTICS PIPELINE RESID** system command. They are mapped by the DFHPIRDS DSECT.

The resource information gives details of various attribute settings of each PIPELINE resource. A total use count for all PIPELINE resources is also available.

Table 155. PIPELINE definitions: resource statistics

DFHSTUP name	Field name	Description
PIPELINE Name	PIR_PIPELINE_NAME	The name of the PIPELINE resource definition. Reset characteristic: not reset
PIPELINE Mode	PIR_PIPELINE_MODE	The operating mode of the pipeline. Reset characteristic: not reset
Configuration file	PIR_CONFIGURATION_FILE	The name of the zFS file that provides information about the message handlers and their configuration. Reset characteristic: not reset
Shelf directory	PIR_SHELF_DIRECTORY	The fully qualified name of the shelf directory for the PIPELINE definition. Reset characteristic: not reset
WSDIR pickup directory	PIR_WSDIR_DIRECTORY	The fully qualified name of the web service binding directory (also known as the pickup directory). Reset characteristic: not reset
PIPELINE use count	PIR_PIPELINE_USE_COUNT	The number of times this PIPELINE resource definition was used to install a web service or to process a web service request. Reset characteristic: reset to zero

Table 155. PIPELINE definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
PIPELINE JSON_JAVA_PARSER	PIR_JSON_JAVA_PARSER	For a JSON PIPELINE resource, specifies if the JSON request message is parsed using Java or from within the CICS pipeline. Reset characteristic: never reset PIR_JSON_JAVA_PARSER is expected to take 1 of 3 values: • pir_json_java_parser_notapplic = 0 pir_json_java_parser_notapplic is set for a PIPELINE that is not for JSON; for example, a SOAP PIPELINE. • pir_json_java_parser_yes = 1 pir_json_java_parser_yes is set when a JSON PIPELINE configuration file contains the attribute `java_parser=yes`. This is the default value. • pir_json_java_parser_no = 2 pir_json_java_parser_no is set when a JSON PIPELINE configuration file contains the attribute `java_parser=no`.
Not in DFHSTUP report	PIR_PIPELINE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	PIR_PIPELINE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	PIR_PIPELINE_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	PIR_PIPELINE_CHANGE_AGENT	Identifies the agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	PIR_PIPELINE_INSTALL_AGENT	Identifies the agent that installed the resource. Reset characteristic: not reset

Table 155. PIPELINE definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PIR_PIPELINE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	PIR_PIPELINE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
PIPELINE message format	PIR_PIPELINE_MSGFORMAT	The message format processed by the PIPELINE. Reset characteristic: not reset

Pipeline totals:

The resource statistics also include a total PIPELINE use count, which shows the total number of times a PIPELINE resource definition was used to install a web service or to process a web service request.

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

PIPELINE definitions: Summary resource statistics

Summary statistics are not available online.

The resource information gives details of various attribute settings of each PIPELINE definition. A total use count for all PIPELINE definitions is also available.

Table 156. PIPELINE definitions: Summary resource statistics

DFHSTUP name	Description
PIPELINE Name	The name of the PIPELINE resource definition.
PIPELINE Mode	The operating mode of the pipeline.

Table 156. PIPELINE definitions: Summary resource statistics (continued)

DFHSTUP name	Description
Configuration file	The name of the z/OS UNIX file that provides information about the message handlers and their configuration.
Shelf directory	The fully qualified name of the shelf directory for the PIPELINE definition.
WSDIR pickup directory	The fully qualified name of the web service binding directory (also known as the pickup directory).
PIPELINE use count	The number of times this PIPELINE resource definition was used to install a web service or to process a web service request.
PIPELINE JSON_JAVA_PARSER	For a JSON PIPELINE resource, specifies if the JSON request message is parsed by using Java or from within the CICS pipeline.

Pipeline Totals:

The summary statistics also include a total PIPELINE use count, which shows the total number of times a PIPELINE resource definition was used to install a web service or to process a web service request.

Program statistics

Program statistics report the resource data collected by the loader for each program.

For public programs, these statistics are mapped by the DFHLDRDS DSECT. For private programs for applications that are deployed on platforms, these statistics are mapped by the DFHLDPDS DSECT. The statistics records for private programs have information about the application for which the program was loaded.

Programs that are defined as application entry points are not identified in the program loader statistics, and only a private program statistics record is produced for them.

Information about Java programs that run in a JVM is not included in the program statistics, because JVM programs are not loaded by CICS. For this information, see "JVM program statistics" on page 583.

Related concepts:

"Interpreting program statistics" on page 650
Average fetch time is an indication of how long it takes MVS to perform a load from the partitioned data set in the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage.

Related reference:

"Programs report" on page 860
The Programs report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

"Program Totals report" on page 864
The Program Totals Report is calculated from data obtained using the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

"DFHRPL and LIBRARY Analysis report" on page 813
The DFHRPL and LIBRARY Analysis report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM**, **EXEC CICS COLLECT STATISTICS PROGRAM** and **EXEC CICS EXTRACT LIBRARY** commands. The statistics data was mapped by the DFHLDRDS and **DFHLDBDS DSECT**.

"Programs by DSA and LPA report" on page 863
The Programs by DSA and LPA report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

"JVM Programs report" on page 842
The JVM Programs report shows information and statistics about Java programs that run in JVM servers or pooled JVMs. This report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** commands. The statistics data is mapped by the **DFHPGRDS DSECT**.

"User Exit Programs report" on page 923
The User Exit Programs report is produced from two tables. This report is produced using the **EXEC CICS INQUIRE EXITPROGRAM** command.

"Global User Exits report" on page 836
The Global User Exits report is produced using the **EXEC CICS INQUIRE EXITPROGRAM** command.

Interpreting program statistics

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

The average for each LIBRARY offset (Lbry ofst) of "Program size" / "Average fetch time". 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.

Programs - Public: Resource statistics

You can retrieve statistics for public programs by using the **EXEC CICS EXTRACT STATISTICS PROGRAM** system command. They are mapped by the DFHLDRDS DSECT.

Program resource statistics for public programs contain the resource data collected by the loader for each public program.

Statistics for public programs are mapped by the DFHLDRDS DSECT. For private programs for applications that are deployed on platforms, these statistics are mapped by the DFHLDPDS DSECT, which has information about the application for which the program was loaded. For details of the DSECT and DFHSTUP report for private programs, see “Programs - Private: Resource statistics” on page 653. Programs that are defined as application entry points are not identified in the program loader statistics.

Table 157. Programs - Public: Resource statistics

DFHSTUP name	Field name	Description
Program name	LDRPNAME	The name of the program. Reset characteristic: not reset
Times used	LDRTU	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	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the static DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	LDRFT	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	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
Lbry ofst	LDRRPLO	The offset into the static DFHRPL or dynamic LIBRARY DD concatenation of the data set from which the program is currently loaded or will be 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. Reset characteristic: not reset
NEWCOPY count	LDRTN	The number of times a NEWCOPY has been requested against this program. Reset characteristic: reset to zero

Table 157. Programs - Public: Resource statistics (continued)

DFHSTUP name	Field name	Description
Program size	LDRPSIZE	The size of the program in bytes, if known (otherwise zero). Reset characteristic: not reset
Times removed	LDRRPC	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
Current Location	LDRLOCN	The location of the current storage resident instance of the program, if any. It has one of the values shown in Table 158. Reset characteristic: not reset
LIBRARY name	LDRLBNM	The name of the LIBRARY from which the program was loaded. Reset characteristic: not reset
LIBRARY Dsname	LDRLBDNM	The name of the data set in the LIBRARY from which the program was loaded. Reset characteristic: not reset

Table 158. Values for Location (LDRLOCN)

DFHSTUP value	DSECT value	Meaning
NONE	LDRNOCO (X'00')	No current copy
CDSA	LDRCDCO (X'01')	Current copy in the CDSA
SDSA	LDRSDCO (X'08')	Current copy in the SDSA
LPA	LDRLPACO (X'03')	Current copy in the LPA
ECDSA	LDRECDCO (X'04')	Current copy in the ECDSA
ESDSA	LDRESDCO (X'09')	Current copy in the ESDSA
ERDSA	LDRERDCO (X'06')	Current copy in the ERDSA
RDSA	LDRRDCO (X'0A')	Current copy in the RDSA

Related information:

EXTRACT STATISTICS in Reference > System programming

Programs - Public: Summary resource statistics

A summary listing of resource statistics for the loader for each public program.

Summary statistics are not available online.

Private programs for applications that are deployed on platforms are reported in a separate summary report. For details of this report, see "Programs - Private: Summary resource statistics" on page 656.

Table 159. Programs - Public: Summary resource statistics

DFHSTUP name	Description
Program name	The name of the program.
Times used	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	The total number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage.
Average fetch time	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	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.
LIBRARY name	The name of the LIBRARY concatenation from which the program was loaded.
LIBRARY Dsname	The name of the data set in the LIBRARY concatenation from which the program was loaded.

Programs - Private: Resource statistics

You can retrieve statistics for private programs for applications that are deployed on platforms by using the **EXEC CICS EXTRACT STATISTICS PROGRAM** system command. They are mapped by the DFHLDPDS DSECT.

Program resource statistics for private programs contain the resource data collected by the loader for each private program for applications deployed on platforms.

Statistics for private programs for applications that are deployed on platforms are mapped by the DFHLDPDS DSECT. For public programs, these statistics are mapped by the DFHLDRDS DSECT. For details of the DSECT and DFHSTUP report for public programs, see "Programs - Public: Resource statistics" on page 650. Programs that are defined as application entry points are not identified in the program loader statistics.

Table 160. Programs - Private: Resource statistics

DFHSTUP name	Field name	Description
Platform	LDP_PLATFORM_NAME	The name of the platform where the application that uses the private programs is deployed. Reset characteristic: not reset
Application	LDP_APPLICATION_NAME	The name of the application that uses the private programs. Reset characteristic: not reset
Major version	LDP_APPL_MAJOR_VER	The major version number of the application that uses the private programs. Reset characteristic: not reset
Minor version	LDP_APPL_MINOR_VER	The minor version number of the application that uses the private programs. Reset characteristic: not reset
Micro version	LDP_APPL_MICRO_VER	The micro version number of the application that uses the private programs. Reset characteristic: not reset
Program name	LDPPNAME	The name of the program. Reset characteristic: not reset
Times used	LDPTU	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	LDPFC	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the static DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Reset characteristic: reset to zero

Table 160. Programs - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	LDPFT	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	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
Lbry ofst	LDPRPLO	The offset into the static DFHRPL or dynamic LIBRARY DD concatenation of the data set from which the program is currently loaded or will be 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. Reset characteristic: not reset
NEWCOPY count	LDPTN	The number of times a NEWCOPY has been requested against this program. Reset characteristic: reset to zero
Program size	LDPPSIZE	The size of the program in bytes, if known (otherwise zero). Reset characteristic: not reset
Times removed	LDPRPC	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
Current Location	LDPLOCN	The location of the current storage resident instance of the program, if any. It has one of the values shown in Table 161 on page 656. Reset characteristic: not reset
LIBRARY name	LDPLBNM	The name of the LIBRARY from which the program was loaded. Reset characteristic: not reset

Table 160. Programs - Private: Resource statistics (continued)

DFHSTUP name	Field name	Description
LIBRARY Dsname	LDPLBDNM	The name of the data set in the LIBRARY from which the program was loaded. Reset characteristic: not reset
Operation	LDP_OPERATION_NAME	For programs that are declared as application entry points, the application operation that is named for the application entry point. Reset characteristic: not reset

Table 161. Values for Location (LDPLOCN)

DFHSTUP value	DSECT value	Meaning
NONE	LDPNOCO (X'00')	No current copy
CDSA	LDPCDCO (X'01')	Current copy in the CDSA
SDSA	LDPSDCO (X'08')	Current copy in the SDSA
LPA	LDPLPACO (X'03')	Current copy in the LPA
ECDSA	LDPECDCO (X'04')	Current copy in the ECDSA
ESDSA	LDPESDCO (X'09')	Current copy in the ESDSA
ERDSA	LDPERDCO (X'06')	Current copy in the ERDSA
RDSA	LDPRDCO (X'0A')	Current copy in the RDSA

Related information:

EXTRACT STATISTICS in Reference > System programming

Programs - Private: Summary resource statistics

A summary listing of resource statistics for the loader for private programs for applications that are deployed on platforms.

Summary statistics are not available online.

Public programs are reported in a separate summary report. For details of this report, see “Programs - Public: Summary resource statistics” on page 653.

Table 162. Programs - Private: Summary resource statistics

DFHSTUP name	Description
Platform	The name of the platform where the application that uses the private programs is deployed.
Application	The name of the application that uses the private programs.
Major version	The major version number of the application that uses the private programs.
Minor version	The minor version number of the application that uses the private programs.
Micro version	The micro version number of the application that uses the private programs.
Program name	The name of the program.
Operation	For programs that are declared as application entry points, the application operation that is named for the application entry point.
Times used	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	The total number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage.
Average fetch time	The average time taken to perform a fetch of the program. The DFHSTUP report expresses this time as *minutes:seconds.decimals*.
NEWCOPY count	The total number of times a NEWCOPY has been requested against this program.
Times removed	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.
LIBRARY name	The name of the LIBRARY concatenation from which the program was loaded.
LIBRARY Dsname	is the name of the data set in the LIBRARY concatenation from which the program was loaded.

Program definition statistics

Program definition statistics report the resource data collected by the Program Manager for each program.

For public programs, these statistics are mapped by the DFHPGDDS DSECT. For private programs for applications that are deployed on platforms, these statistics are mapped by the DFHPGEDS DSECT. The statistics records for private program definitions have information about the application for which the program was defined.

Program definitions that are declared as application entry points are identified by a field in the DFHPGDDS and DFHPGRDS DSECTs for public program definitions and JVM programs, and by a field in the DFHPGEDS and DFHPGPDS DSECTs for private program definitions and JVM programs. When interval statistics, end-of-day statistics, requested statistics, requested reset statistics, or unsolicited statistics are produced for a program definition that is declared as an application entry point, two statistics records are written, one mapped by the DSECT for public resources, and one mapped by the DSECT for private resources.

Program definitions - Public: Resource statistics

You can retrieve statistics for public program definitions by using the **EXEC CICS EXTRACT STATISTICS PROGRAMDEF** system command. They are mapped by the DFHPGDDS DSECT.

Program definition resource statistics for public program definitions contain the resource data collected by the Program Manager for each program.

Statistics for public program definitions are mapped by the DFHPGDDS DSECT. For private program definitions for applications that are deployed on platforms, these statistics are mapped by the DFHPGEDS DSECT, which has information about the application for which the program was defined. For details of the DSECT and DFHSTUP report for private program definitions, see “Program definitions - Private: Resource statistics” on page 662.

Programs that are defined as application entry points are identified by the PGD_PROGRAM_ENTRYPOINT field. Both public and private statistics records are written for these programs, mapped once by each DSECT.

Table 163. Program definitions - Public: resource statistics

DFHSTUP name	Field name	Description
Program Name	PGD_PROGRAM_NAME	The name of the program. Reset characteristic: not reset
Type	PGD_PROGRAM_TYPE	The type of module. Reset characteristic: not reset
EXEC key	PGD_PROGRAM_EXEC_KEY	The access key in which the program will run. Reset characteristic: not reset
Data loc	PGD_PROGRAM_DATA_LOC	The storage location that the program can accept. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_EXECUTION_SET	Whether the module is restricted to the distributed program link subset of the CICS API. EXECUTIONSET applies only to executable programs, and governs the API only when a program is invoked locally. Reset characteristic: not reset

Table 163. Program definitions - Public: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PGD_PROGRAM_LANG_DEDUCED	The language of the module. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_LANGUAGE	The program language as defined in the LANGUAGE attribute of the program definition. Reset characteristic: not reset
Runtime	PGD_PROGRAM_RUNTIME_ENV	The runtime environment of the program. Reset characteristic: not reset
Concurrency	PGD_PROGRAM_CONCURRENCY	The concurrency attribute (QUASIRENT, THREADSAFE, or REQUIRED) of the installed program definition. Reset characteristic: not reset
API	PGD_PROGRAM_API	The API attribute (CICS or OPEN) of the installed program definition Reset characteristic: not reset
Remote	PGD_PROGRAM_REMOTE	Whether, if the program is the subject of a program-link request, it can be statically routed. Reset characteristic: not reset
Dynamic	PGD_PROGRAM_DYNAMIC	Whether, if the program is the subject of a program-link request, it can be dynamically routed. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_JVM	Whether the program is a Java program that must run in a JVM server. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_ENTRYPOINT	Whether the program is defined as an application entry point for an application deployed on a platform. Reset characteristic: not reset

Table 163. Program definitions - Public: resource statistics (continued)

DFHSTUP name	Field name	Description
Remote Name	PGD_PROGRAM_REMOTE_NAME	For programs only, the name by which the module is known in the CICS region named in the Remote System field, and only to those defined to be remote. Reset characteristic: not reset
Remote Tran	PGD_PROGRAM_TRAN_ID	For programs only, the name of the transaction under which this module, which must be a program, runs remotely; that is, the transaction identifier that the remote region assigns to the task created there to execute it when a task in the local region LINKs to it. Reset characteristic: not reset
Remote System	PGD_PROGRAM_REMOTE_SYSID	For programs only, the name of the CICS region in which the module is defined. It applies only to programs, and only to those defined to be remote. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_JVMSERVER	For a Java program, the name of the JVM server in which this Java program runs. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_CHANGE_USERID	The user ID that ran the CHANGEAGENT. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset

Table 163. Program definitions - Public: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PGD_PROGRAM_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	PGD_PROGRAM_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Program definitions - Public: summary resource statistics

A summary listing of resource statistics for all public program definitions.

Summary resource statistics are not available online.

Private program definitions for applications that are deployed on platforms are reported in a separate summary report. For details of this report, see “Program definitions - Private: summary resource statistics” on page 666. Programs that are defined as application entry points appear in both the public and private resource summary reports.

Table 164. Program definitions - public: summary resource statistics

DFHSTUP name	Description
Program Name	The name of the program.
Type	The type of module.
Concurrency	The concurrency attribute of the installed program definition.
API	The API attribute (CICS or OPEN) of the installed program definition
EXEC Key	The access key in which the program runs.
Data Loc	The storage location that the program can accept.
Language Deduced	The language of the program.
Runtime Environment	The runtime environment of the program.
Remote	Whether, if the program is the subject of a program-link request, it can be statically routed.

Table 164. Program definitions - public: summary resource statistics (continued)

DFHSTUP name	Description
Dynamic	Whether, if the program is the subject of a program-link request, it can be dynamically routed.
Remote Name	For programs only, the name by which the module is known in the CICS region named in the Remote System field, and only to those defined to be remote.
Remote Tran	For programs only, the name of the transaction under which this module, which must be a program, runs remotely (that is, the transaction identifier that the remote region assigns to the task created there to run it when a task in the local region LINKs to it).
Remote System	For programs only, the name of the CICS region in which the module is defined. It applies only to programs, and only to those defined to be remote.

Program definitions - Private: Resource statistics

You can retrieve statistics for private program definitions for applications that are deployed on platforms by using the **EXEC CICS EXTRACT STATISTICS PROGRAMDEF** system command. They are mapped by the DFHPGEDS DSECT.

Program definition resource statistics for private program definitions contain resource data, collected by the Program Manager, for the private programs for applications that are deployed on platforms.

Statistics for private program definitions for applications that are deployed on platforms are mapped by the DFHPGEDS DSECT. For public program definitions, these statistics are mapped by the DFHPGDDS DSECT. For details of the DSECT and DFHSTUP report for public program definitions, see “Program definitions - Public: Resource statistics” on page 658.

Programs that are defined as application entry points are identified by an application operation being named in the PGE_PROGRAM_OPERATION_NAME field. Both public and private statistics records are written for these programs, mapped once by each DSECT.

The DFHSTUP report shows the private programs for each application that is deployed on a platform. For programs that are declared as application entry points, the report shows the application operation that is named for the application entry point.

Table 165. Program definitions - Private: resource statistics

DFHSTUP name	Field name	Description
Platform	PGE_PROGRAM_PLATFORM_NAME	The name of the platform where the application that uses the private programs is deployed. Reset characteristic: not reset
Application	PGE_PROGRAM_APPLICATION_NAME	The name of the application that uses the private programs. Reset characteristic: not reset

Table 165. Program definitions - Private: resource statistics (continued)

DFHSTUP name	Field name	Description
Major version	PGE_PROGRAM_APPL_MAJOR_VER	The major version number of the application that uses the private programs. Reset characteristic: not reset
Minor version	PGE_PROGRAM_APPL_MINOR_VER	The minor version number of the application that uses the private programs. Reset characteristic: not reset
Micro version	PGE_PROGRAM_APPL_MICRO_VER	The micro version number of the application that uses the private programs. Reset characteristic: not reset
Program Name	PGE_PROGRAM_NAME	The name of the private program. Reset characteristic: not reset
Type	PGE_PROGRAM_MODULE_TYPE	The type of module. Reset characteristic: not reset
EXEC key	PGE_PROGRAM_EXEC_KEY	The access key in which the program will run. Reset characteristic: not reset
Data loc	PGE_PROGRAM_DATA_LOC	The storage location that the program can accept. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_EXECUTION_SET	Whether the module is restricted to the distributed program link subset of the CICS API. EXECUTIONSET applies only to executable programs, and governs the API only when a program is invoked locally. Reset characteristic: not reset
Language Deduced	PGE_PROGRAM_LANG_DEDUCED	The language of the module. Reset characteristic: not reset

Table 165. Program definitions - Private: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PGE_PROGRAM_LANGUAGE	The program language as defined in the LANGUAGE attribute of the program definition. Reset characteristic: not reset
Runtime	PGE_PROGRAM_RUNTIME_ENV	The runtime environment of the program. Reset characteristic: not reset
Concurrency	PGE_PROGRAM_CONCURRENCY	The concurrency attribute (QUASIRENT, THREADSAFE, or REQUIRED) of the installed program definition. Reset characteristic: not reset
API	PGE_PROGRAM_API	The API attribute (CICS or OPEN) of the installed program definition Reset characteristic: not reset
Remote	PGE_PROGRAM_REMOTE	Whether, if the program is the subject of a program-link request, it can be statically routed. Reset characteristic: not reset
Dynamic	PGE_PROGRAM_DYNAMIC	Whether, if the program is the subject of a program-link request, it can be dynamically routed. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_JVM	Whether the program is a Java program that must run in a JVM server. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_ENTRYPOINT	Whether the program is defined as an application entry point for an application deployed on a platform. Reset characteristic: not reset

Table 165. Program definitions - Private: resource statistics (continued)

DFHSTUP name	Field name	Description
Remote Name	PGE_PROGRAM_REMOTE_NAME	For programs only, the name by which the module is known in the CICS region named in the Remote System field, and only to those defined to be remote. Reset characteristic: not reset
Remote Tran	PGE_PROGRAM_TRAN_ID	For programs only, the name of the transaction under which this module, which must be a program, runs remotely; that is, the transaction identifier that the remote region assigns to the task created there to execute it when a task in the local region LINKs to it. Reset characteristic: not reset
Remote System	PGE_PROGRAM_REMOTE_SYSID	For programs only, the name of the CICS region in which the module is defined. It applies only to programs, and only to those defined to be remote. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_JVMSERVER	For a Java program, the name of the JVM server in which this Java program runs. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_CHANGE_USERID	The user ID that ran the CHANGEAGENT. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset

Table 165. Program definitions - Private: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PGE_PROGRAM_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	PGE_PROGRAM_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset
Operation	PGE_PROGRAM_OPERATION_NAME	For programs that are declared as application entry points, the application operation that is named for the application entry point. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Program definitions - Private: summary resource statistics

A summary listing of resource statistics for all private program definitions for applications that are deployed on platforms.

Summary resource statistics are not available online.

Public program definitions are reported in a separate summary report. For details of this report, see "Program definitions - Public: summary resource statistics" on page 661. Programs that are defined as application entry points appear in both the public and private resource summary reports.

Table 166. Program definitions - private: summary resource statistics

DFHSTUP name	Description
Platform	The name of the platform where the application that uses the private programs is deployed.
Application	The name of the application that uses the private programs.

Table 166. Program definitions - private: summary resource statistics (continued)

DFHSTUP name	Description
Major version	The major version number of the application that uses the private programs.
Minor version	The minor version number of the application that uses the private programs.
Micro version	The micro version number of the application that uses the private programs.
Program Name	The name of the private program.
Operation	For programs that are declared as application entry points, the application operation that is named for the application entry point.
Type	The type of module.
Concurrency	The concurrency attribute of the installed program definition.
API	The API attribute (CICS or OPEN) of the installed program definition
EXEC Key	The access key in which the program runs.
Data Loc	The storage location that the program can accept.
Language Deduced	The language of the program.
Runtime Environment	The runtime environment of the program.
Remote	Whether, if the program is the subject of a program-link request, it can be statically routed.
Dynamic	Whether, if the program is the subject of a program-link request, it can be dynamically routed.
Remote Name	For programs only, the name by which the module is known in the CICS region named in the Remote System field, and only to those defined to be remote.
Remote Tran	For programs only, the name of the transaction under which this module, which must be a program, runs remotely (that is, the transaction identifier that the remote region assigns to the task created there to run it when a task in the local region LINKs to it).
Remote System	For programs only, the name of the CICS region in which the module is defined. It applies only to programs, and only to those defined to be remote.

Recovery manager statistics

Recovery manager statistics detail the sync point activity of all the transactions in the system. From these statistics, you can assess the impact of shunted UOWs (units of work that suffered an indoubt failure and are waiting for resynchronization with their recovery coordinator, or for the problem with the resources to be resolved).

Shunted UOWs still hold locks and enqueues until they are resolved. Statistics are available on any forced resolutions of shunted UOWs to help assess whether any integrity exposures have been introduced. The current activity and the activity since the last reset is available.

Related reference:

"Recovery Manager report" on page 866
The Recovery Manager report is produced using the **EXEC CICS EXTRACT STATISTICS RECOVERY** command. The statistics data is mapped by the **DFHRMGDS DSECT**.

Recovery manager: Global statistics

You can retrieve recovery manager statistics by using the **EXEC CICS EXTRACT STATISTICS RECOVERY** system command. They are mapped by the DFHRMGDS DSECT.

Table 167. Recovery manager: Global statistics

DFHSTUP name	Field name	Description
Total number of syncpoints (forward)	RMGSYFWD	is the total number of syncpoint requests to commit forward. Reset characteristic: reset to zero
Total number of syncpoints (backward)	RMGSYBWD	is the total number of syncpoint requests to commit backward (for example, EXEC CICS SYNCPOINT ROLLBACK). Reset characteristic: reset to zero
Total number of resynchronizations	RMGRESYN	is the total number of resynchronization requests. Reset characteristic: reset to zero
Total shunted UOWs for indoubt failure	RMGTSHIN	is the total number of units of work that lost connection to their recovery coordinator during syncpoint processing and had to be shunted for indoubt failure, but have now completed. Note that this value does not include those units of work that are currently shunted for indoubt failure. Reset characteristic: reset to zero
Total time shunted for indoubt failure	RMGTSHTI	is the total time (STCK) that the units of work shunted for indoubt failure (RMGTSHIN) spent waiting in this condition, but have now completed. Note that this value does not include those units of work that are currently shunted for indoubt failure. Reset characteristic: reset to zero
Total shunted UOWs for commit/backout failure	RMGTSHRO	is the total number of units of work that had to be shunted for commit/backout failure because a local resource manager could not perform commit/backout processing at this time on behalf of the UOW during syncpoint, but have now completed. Note that this value does not include those units of work that are currently shunted for commit/backout failure. Reset characteristic: reset to zero
Total time shunted for commit/backout failure	RMGTSHTR	is the total time (STCK) that the units of work shunted for commit/backout (RMGTSHRO) failures spent waiting in this condition, but have now completed. Note that this value does not include those units of work that are currently shunted for commit/backout failure. Reset characteristic: reset to zero

Table 167. Recovery manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Current shunted UOWs for indoubt failure	RMGCSHIN	is the current number of units of work that lost the connection to their recovery coordinator during syncpoint processing, and have been shunted for indoubt failure. Reset characteristic: reset to zero
Current time shunted for indoubt failure	RMGCSHTI	is the total time (STCK) that the units of work currently shunted for indoubt failure (RMGCSHIN) have been waiting in this condition so far. Reset characteristic: reset to zero
Current shunted UOWs for resource failure	RMGCHSHR	is the current number of units of work that have been shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at this time on behalf of the UOW during syncpoint Reset characteristic: reset to zero
Current time shunted for resource failure	RMGCSHTR	is the total time (STCK) that the units of work currently shunted for commit/backout (RMGCHSHR) failures have been waiting in this condition so far. Reset characteristic: reset to zero

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

DFHSTUP name	Field name	Description
Total forces of indoubt action by trandef	RMGIAFTR	is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specified that they could not wait indoubt. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

DFHSTUP name	Field name	Description
Total forces of indoubt action by timeout	RMGIAFTI	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Total forces of indoubt action by operator	RMGIAFOP	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, through a CEMT, or EXEC CICS, SET UOW command forced a resolution. The UOWs would have committed or backed out according to the command option, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Total forces of indoubt action by no wait	RMGIAFNW	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because a local resource owner or connected resource manager used by the UOW was unable to wait indoubt. The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. See the following section on no support for indoubt waiting breakdown. Reset characteristic: reset to zero
Total forces of indoubt action by other	RMGIAFOT	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because of reasons other than those already referenced in this table (for example, a cold start of the coordinator, level of RMI adapter modification, and resynchronization errors). The UOWs would have committed or backed out according to the transaction definition indoubt action attribute, regardless of the actions specified or taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero

The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field RMGIAFNW. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.

DFHSTUP name	Field name	Description
–Indoubt action forced by TD queues	RMGNWTD	is the number of UOW forces that occurred because the UOW uses a recoverable transient data queue defined with an indoubt attribute of WAIT=NO. Reset characteristic: reset to zero
–Indoubt action forced by LU61 connections	RMGNW61	is the number of UOW forces that occurred because the UOW uses an LU6.1 intersystem link, which cannot support indoubt waiting. Note that if an LU6.1 intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Syncpoint exchanges in the *CICS Intercommunication Guide*. Reset characteristic: reset to zero
–Indoubt action forced by MRO connections	RMGNWMRO	is the number of UOW forces that occurred because the UOW uses an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting. Note that if an MRO intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Syncpoint exchanges in the *CICS Intercommunication Guide*. Reset characteristic: reset to zero
–Indoubt action forced by RMI exits (TRUEs)	RMGNWRMI	is the number of UOW forces that occurred because the UOW uses an RMI that declared an interest in syncpoint but could not support indoubt waiting. Note that if an RMI intersystem link can operate as last agent in syncpoint processing the lack of waiting ability is immaterial. For more details about last agent processing, see Syncpoint exchanges in the *CICS Intercommunication Guide*. Reset characteristic: reset to zero
–Indoubt action forced by others	RMGNWOTH	is the number of UOW forces that occurred because the UOW uses recoverable facilities other than already referenced in this table (for example, terminal RDO), which invalidate the ability to support indoubt waiting. Reset characteristic: reset to zero

The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field RMGIAFNW. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.

DFHSTUP name	Field name	Description
–Total number of indoubt action mismatches	RMGIAMIS	is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option or operator override (as detailed in the fields already referenced in this table), and on so doing detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Recovery manager: Summary global statistics

Recovery manager summary statistics are not available online.

Table 168. Recovery manager: Summary global statistics

DFHSTUP name	Description
Total number of syncpoints (forward)	is the total number of syncpoint requests to commit forward.
Total number of syncpoints (backward)	is the total number of syncpoint requests to commit backward. For example, EXEC CICS SYNCPOINT ROLLBACK.
Total number of resynchronizations	is the total number of resynchronization requests.
Total shunted UOWs for indoubt failure	is the total number of UOWs that have lost connection to their recovery coordinator during syncpoint processing, had to be shunted for indoubt failure, but have now completed.
Total time shunted for indoubt failure	is the total time (STCK) that the UOWs shunted for indoubt failure ('Total number of shunts for indoubt failure) spent waiting in this condition.
Total shunted UOWs for commit/backout failure	is the total number of UOWs that had to be shunted for commit/backout failure because a local resource manager was not able to perform commit/backout processing at that time, but have now completed.
Total time shunted for commit/backout failure	is the total time (STCK) that the UOWs shunted for commit/ backout ('Total UOWs shunted for commit/backout failure) failures waited in this condition, but have now completed.
Outstanding shunted UOWs for indoubt failure	is the current number of UOWs that have been shunted for indoubt failure because the connection to their recovery coordinator during syncpoint processing was lost.
Outstanding time shunted for indoubt failure	is the total time (STCK) that the UOWs currently shunted for indoubt failure spent waiting in this condition so far.
Outstanding shunted UOWs for resource failure	is the current number of UOWs that have been shunted for commit/ backout failure because a local resource manager was unable to perform commit/backout processing at that time on behalf of the UOW.
Outstanding time shunted for resource failure	is the total time (STCK) that the UOWs currently shunted for commit/backout failures have been waiting in this condition so far.

The following fields detail the reasons why UOWs may have introduced integrity exposures because they were forced to complete prematurely. The UOWs were not allowed to shunt, not capable of shunting, or forced to terminate a shunt, regardless of the outcome.

Table 168. Recovery manager: Summary global statistics (continued)

DFHSTUP name	Description
Total forces of indoubt action by trandef	is the total number of UOWs that were forced to complete syncpoint processing, despite losing the connection to the recovery coordinator, because their transaction definition specified that they could not wait indoubt.
Total forces of indoubt action by timeout	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator, because their transaction definition wait for indoubt timeout value was exceeded.
Total forces of indoubt action by operator	is the total number of shunted indoubt UOWs that were forced to complete syncpoint processing, although still unconnected to the recovery coordinator because the operator (CEMT) forced a resolution.
Total forces of indoubt action by no wait	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because a local resource owner or connected resource manager that the UOW used was unable to wait indoubt. Further details are provided by the section in the table titled, No support for indoubt waiting breakdown.
Total forces of indoubt action by other	is the total number of UOWs that were forced to complete syncpoint processing, despite having the ability to wait indoubt, because of reasons other than those already referenced in the table (for example, a cold start of the coordinator, level of RMI adapter modification, and resynchronization errors).
No support for indoubt waiting breakdown	
	The following fields further detail the reasons why a UOW did not have the ability to wait indoubt (shunt) at the time of indoubt failure (lost coordinator), and are breakdowns of the field 'Total forces of indoubt action by no wait'. This is because the UOW uses either recoverable local resources, recoverable resources across intersystem links, or external resource managers (RMI), which do not have the ability to wait indoubt. As a result of a resolution of a UOW being forced for this reason, integrity exposures may occur.
–Indoubt action forced by TD queues	is the number of UOW forces that occurred because the UOW was using a recoverable transient data queue defined with an indoubt attribute of WAIT=NO.
–Indoubt action forced by LU61 connections	is the number of UOW forces that occurred because the UOW used an LU6.1 intersystem link, which cannot support indoubt waiting.
–Indoubt action forced by MRO connections	is the number of UOW forces that occurred because the UOW used an MRO intersystem link to a downlevel CICS region, which cannot support indoubt waiting.
–Indoubt action forced by RMI exits (TRUEs)	is the number of UOW forces that occurred because the OUW used an RMI that declared an interest in syncpoint but could not support indoubt waiting.
–Indoubt action forced by others	is the number of UOW forces that occurred because the UOW used recoverable facilities other than those already referenced in the table, for example, terminal RDO, which invalidates the ability to support indoubt waiting.
Total number of indoubt action mismatches	is the total number of UOWs that were forced to resolve using an indoubt action attribute, whether by definition, option, or operator override (as detailed in this table), and detected an indoubt action attribute mismatch with a participating system or RMI. For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Shared temporary storage queue server statistics

Shared temporary storage queue server statistics are provided by the AXM page pool management routines for the pools AXMPGANY and AXMPGLOW.

Shared TS queue server: coupling facility statistics

For queues that do not exceed 32K bytes, the data is included in the queue index; otherwise, it is stored as a separate list.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The statistics are described in detail in the DFHXQS1D data area. The individual fields have the following meanings.

Table 169. Shared TS queue server: coupling facility statistics

Statistic name	Field	Description
Structure	S1PREF	First part of structure name
Structure	S1POOL	Poolname part of structure name
Structure	S1CNPREF	Prefix for connection name
Structure	S1CNSYSN	Own MVS system name from CVTSNAME
Structure: Size	S1SIZE	Current allocated size of the list structure.
Structure: Elem size	S1ELEMLN	Data element size, fullword, used for the structure.
Structure: Max size	S1SIZEMX	Maximum size to which this structure could be altered.
Lists: Total	S1HDRS	Maximum number of list headers
Lists: Control	S1HDRSCT	Headers used for control lists
Lists: Data	S1HDRSQD	Headers available for queue data
Lists: In use	S1USEDCT	Number of entries on used list
Lists: Max used	S1USEDHI	Highest number of entries on used list
Entries: In Use	S1ENTRCT	Number of entries currently in use.
Entries: Max Used	S1ENTRHI	Maximum number in use (since last reset).
Entries: Min Free	S1ENTRLO	Minimum number of free entries (since last reset).
Entries: Total	S1ENTRMX	Total data entries in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
Entries	S1FREECT	Number of entries on free list
Entries	S1ENTRRT	Entry size of entry to element ratio
Entries	S1FREEHI	Highest number of entries on free list
Elements: In use	S1ELEMCT	Number of elements currently in use.
Elements: Max used	S1ELEMHI	Maximum number in use (since last reset).
Elements: Min free	S1ELEMLO	Number of elements currently free (total minus used).
Elements: Total	S1ELEMMX	Total data elements in the currently allocated structure. (Obtained at connection time, may be updated by ALTER).
Elements	S1ELEMPW	Data element size, power of 2, used for the structure.
Elements	S1ELEMPE	Maximum number of elements per entry (for 32K)
Elements	S1ELEMRT	Element size of entry to element ratio.
Queues: Current	S1INDXCT	Number of queues currently in existence.
Queues: Highest	S1INDXHI	Highest number of queues at any time (since last reset).
Index access counts: Wrt adjs	S1WRACT	Number of index writes to update adjunct area only. (This area contains the read cursor for small queues and the queue status including last used data).
Index access counts: Inquires	S1INQCT	Inquire on queue index entry
Index access counts: Reads	S1RDQCT	Read queue index entry
Index access counts: Writes	S1WRQCT	Write queue index entry.
Index access counts: Deletes	S1DLQCT	Delete queue index entry.

Table 169. Shared TS queue server: coupling facility statistics (continued)

Statistic name	Field	Description
index access counts: Rereads	S1RRQCT	Number of index data reads which had to be repeated because the data was larger than the default data transfer size.
Data access counts: Creates	S1CRLCT	Number of times a separate data list was created.
Data access counts: Writes	S1WRLCT	Number of queue writes (new or update) for list data.
Data access counts: Reads	S1RDLCT	Number of list data reads.
Data access counts: Deletes	S1DLLCT	Delete list (1 per overall delete).
Data access counts: Rereads	S1RRLCT	Number of list data reads which had to be repeated because the data was larger than the default data transfer size.
Data access counts: Rewrites	S1RWLCT	Rewrite list entry.
Data access counts:	S1INLCT	Inquire on list entry
Response counts: Asynch	S1ASYCT	Number of asynchronous requests.
Response counts: Unavail	S1RSP9CT	Structure temporarily unavailable, for example during rebuild.
Response counts: Normal	S1RSP1CT	Number of normal responses.
Response counts: Timeout	S1RSP2CT	Request timed out by the CF and should be restarted.
Response counts: Not fnd	S1RSP3CT	Specified entry (queue or item) was not found.
Response counts: Vers chk	S1RSP4CT	A version check failed for an entry being updated, indicating another task had updated it first.
Response counts: List chk	S1RSP5CT	A list authority comparison failed, usually indicating big queue was deleted.
Response counts: List full	S1RSP6CT	Maximum list key reached, indicating max queue size or max queues reached depending on list.
Response counts: Str full	S1RSP7CT	The list structure is out of space.
Response counts: I/O err	S1RSP8CT	An IXLLIST return code occurred other than those already referenced.

Shared TS queue server: buffer pool statistics

These statistics are for the queue index buffer pool, which is used to read and write queue index entries plus the associated data if the total queue size does not exceed 32K bytes.

Buffers containing recently accessed queue index entries are added to a least recently used chain. This means that if another request for the same queue arrives shortly afterward, it may be possible to optimize the processing based on the assumption that the copy in the buffer is probably already correct. If all other buffers are in use, a request for a new buffer will discard the contents of the least recently used buffer and reuse the storage as a free buffer. The queue server does not use some of the AXM management functions (such as KEEP or PURGE) so those counters will be zero. These fields describe the current state of the buffer pool.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The statistics are described in detail in the DFHXQS2D data area. The individual fields have the following meanings:

Table 170. Shared TS queue server: buffer pool statistics

Statistic name	Field	Description
Buffers: Total	S2BFQTY	Number of buffers in the pool.
Buffers: Max used	S2BFENTH	Highest number ever used (not affected by reset).
Buffers: Active	S2BFACTS	Buffers currently in use.
Buffers: On LRU	S2BFLRUS	Buffers with valid contents on LRU chain to allow reuse.
Buffers: Empty	S2BFEMPS	Buffers previously used but now empty.
Requests: Gets	S2BFGETS	Requests to get a buffer.
Requests: Puts	S2BFPUTS	Put back buffer with valid contents
Requests: Keep	S2BFKEPS	Keeps (put back buffer with modified contents).
Requests: Free	S2BFFRES	Requests to put back a buffer as empty.
Requests: Purges	S2BFPURS	Request to discard contents of a previously valid buffer.
Results (Get): Got hit	S2BFHITS	Buffer requests that found a valid buffer.
Results (Get): Got free	S2BFGFRS	Buffer requests that used a free buffer.
Results (Get): Got new	S2BFGNWS	Buffer requests that obtained a buffer not previously used.
Results (Get): Got LRU	S2BFGLRS	Buffer requests that discarded and reused the oldest valid buffer.
Results (Get): No buf	S2BFGNBS	Buffer requests that returned no buffer.
Error: Not freed	S2BFFNOS	A request tried to release a buffer it did not own. (This can occur during error recovery).
Error: No purge	S2BFPNFS	A purge request did not find a matching buffer.
Error: Not owned	S2BFPNOS	A purge request hit a buffer owned by another task.
Wait: Pool lock	S2BFPWTS	Waits on buffer pool lock.
Wait: Buf lock	S2BFLWTS	GET wait on buffer lock.

Shared TS queue server: storage statistics

These statistics are for the named storage page pool produced since the most recent statistics.

Storage in the AXMPGANY and AXMPGLOW pools is allocated in multiples of 4K pages on a 4K boundary. The most frequent use is for segments of LIFO stack storage. Storage is initially allocated from the pool using a bit map. For faster allocation, free areas are not normally returned to the pool but are added to a vector of free chains depending on the size of the free area (1 to 32 pages). When storage is being acquired, this vector is checked before going to the pool bit map. If there are no free areas of the right size and there is not enough storage remaining in the pool, free areas in the vector are put back into the pool, starting from the smallest end, until a large enough area has been created. This action appears as a compress attempt in the statistics. If there is still insufficient storage to satisfy the request, the request fails.

These statistics are for the named storage page pool produced since the most recent statistics (if any). Each of the storage statistics is shown in kilobytes and as a percentage of the total size.

Reset characteristics: these statistics are produced by a separate server address space, not by CICS. Following a reset, these fields are reset by the server, not CICS. As a general rule, high and low watermarks (max, min and highest, lowest) are reset to current, counts are reset to zero.

The statistics are described in detail in the DFHXQS3D data area.

Table 171. Temporary storage data sharing: usage statistics. **LOC=ANY storage pool statistics**

Statistic name	Field	Description
Name	S3ANYNAM	Name of the storage pool AXMPGANY.
Size	S3ANYSIZ	The total size of the storage pool.
Address	S3ANYPTR	Address of storage pool area.
Total pages	S3ANYMX	Total pages in the storage pool.
In Use	S3ANYUS	The number of pages currently in use.
Free	S3ANYFR	The number of pages within the pool that are currently free.
Min Free	S3ANYLO	The lowest number of pages that have been free (since reset).
Gets	S3ANYRQG	The number of storage GET requests.
Frees	S3ANYRQF	The number of requests to release storage within the pool.
Fails	S3ANYRQS	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Retries	S3ANYRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.

LOC=BELOW storage pool statistics

Statistic name	Field	Description
Name	S3LOWNAM	Name of the storage pool AXMPGLOW.
Size	S3LOWSIZ	The total size of the storage pool.
Address	S3LOWPTR	Address of the storage pool area.
Total pages	S3LOWMX	Total pages in the storage pool.
In Use	S3LOWUS	Number of used pages in the storage pool
Free	S3LOWFR	The number of pages within the pool that are currently free.
Min Free	S3LOWLO	The lowest number of pages that have been free.
Gets	S3LOWRQG	The number of requests to obtain storage within the pool.
Frees	S3LOWRQF	The number of requests to release storage within the pool.
Fails	S3LOWRQS	The number of times that a storage request was unable to obtain the requested amount of storage even after a retry.
Retries	S3LOWRQC	The number of times that a storage request initially failed and was retried after merging any adjacent small free areas to form larger areas.

Statistics domain statistics

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

Statistics domain: Global statistics

You can retrieve statistics domain statistics by using the **EXEC CICS EXTRACT STATISTICS STATS** system command. They are mapped by the DFHSTGDS DSECT.

Table 172. Statistics domain: Global statistics

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.
Number of 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
Number of SMF writes suppressed	STGSMFS	is the number of SMF writes for statistics records that were suppressed by the global user exit (XSTOUT). Reset characteristic: reset to zero
Number of SMF errors	STGSMFE	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
Number of INT statistics records	STGINTR	is the number of SMF writes for interval (INT) statistics records. Reset characteristic: reset to zero
Number of EOD statistics records	STGEODR	is the number of SMF writes for end-of-day (EOD) statistics records. Reset characteristic: reset to zero
Number of USS statistics records	STGUSSR	is the number of SMF writes for unsolicited (USS) statistics records. Reset characteristic: reset to zero

Table 172. Statistics domain: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of REQ statistics records	STGREQR	is the number of SMF writes for requested (REQ) statistics records. Reset characteristic: reset to zero
Number of RRT statistics records	STGRRTR	is the number of SMF writes for requested reset (RRT) statistics records. Reset characteristic: reset to zero
Statistics CICS Start Date and Time	STGCSTRT	is the date and time at which the CICS statistics domain was initialized. The DFHSTUP report expresses the date and time as mm/dd/yyyy and hh:mm:ss; however, the DSECT field contains the date and time as a store clock (STCK) value. Reset characteristic: not reset
Statistics Last Reset Date and Time	STGLRT	is the date and time at which the statistics values were last reset. The DFHSTUP report expresses the date and time as mm/dd/yyyy and hh:mm:ss; however, the DSECT field contains the date and time as a store clock (STCK) value. Reset characteristic: reset to current
Statistics Interval	STGINTVL	is the current statistics recording interval. This is the STATINT value specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET STATISTICS INTERVAL(4-byte packed decimal data-area)** command. Reset characteristic: not reset
Statistics End-of-Day Time	STGEODT	is the current statistics end-of-day time. This is the STATEOD value specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET STATISTICS ENDOFDAY(4-byte packed decimal data-area)** command. Reset characteristic: not reset
Statistics Recording	STGSTRCD	is the current setting for interval statistics recording. This is the STATRCD setting specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET STATISTICS RECORDING(cvda)** command. Reset characteristic: not reset

Table 172. Statistics domain: Global statistics (continued)

DFHSTUP name	Field name	Description
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.

Related information:

EXTRACT STATISTICS in Reference > System programming

Statistics domain: Summary global statistics

Statistics domain summary global statistics are not available online.

Table 173. Statistics domain: Summary global statistics

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.
Total number of SMF writes suppressed	is the total number of SMF writes for statistics records that were suppressed by the global user exit (XSTOUT).
Total number of SMF errors	is the total number of non-OK responses from the request to write a record to SMF.
Total number of INT statistics records	is the total number of SMF writes for interval (INT) statistics records.
Total number of EOD statistics records	is the total number of SMF writes for end-of-day (EOD) statistics records.
Total number of USS statistics records	is the total number of SMF writes for unsolicited (USS) statistics records.
Total number of REQ statistics records	is the total number of SMF writes for requested (REQ) statistics records.
Total number of RRT statistics records	is the total number of SMF writes for requested reset (RRT) statistics records.
Statistics Interval	is the last statistics recording interval (STATINT) value that was specified in the SIT, or as an override, or changed dynamically.

Table 173. Statistics domain: Summary global statistics (continued)

DFHSTUP name	Description
Statistics End-of-Day Time	is the last statistics end-of-day time (STATEOD) value that was specified in the SIT, or as an override, or changed dynamically.
Statistics Recording	is the last setting for interval statistics recording (STATRCD) setting that was specified in the SIT, or as an override, or changed dynamically.

Storage manager statistics

These statistics are produced to aid all aspects of storage management.

Note that the terms 'DSA' (dynamic storage area), and 'pagepool', are interchangeable.

Related concepts:

"Interpreting storage manager statistics"
You can use the "Times went short on storage", "Times request suspended", and "Times cushion released" statistics to assess whether there is sufficient storage.

Related reference:

"Storage reports" on page 867
There are five storage reports. The storage reports provide information about the use of MVS and CICS virtual storage. There are separate reports for storage below 16 MB, storage above 16 MB but below 2 GB, and storage above 2 GB.

"Storage - Domain Subpools reports" on page 881
The storage subpool reports provide statistics about CICS domain and task storage subpool allocations and use.

"Storage - Program Subpools report" on page 884
The Storage Subpools Report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command. The statistics data is mapped by the **DFHSMDDS DSECT**.

Interpreting storage manager statistics

You can use the "Times went short on storage", "Times request suspended", and "Times cushion released" statistics to assess whether there is sufficient storage.

As free storage reduces towards a short-on-storage condition, dynamic program storage compression (DPSC) progressively releases programs that are not in use. However, short-on-storage conditions can still occur and are reported in the "Times went short on storage" statistic. If this value is above zero, consider increasing the size of the dynamic storage area. Alternatively, consider using the maximum tasks (MXT) and transaction class (MAXACTIVE) limits to constrain the virtual storage of your system.

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 statistics, the "Current elem stg" statistic is the number of bytes used, while the "Current page stg" statistic is the number of pages containing one or more of these bytes.

Storage manager: Domain subpools statistics

You can retrieve domain subpool statistics by using the **EXEC CICS EXTRACT STATISTICS STORAGE** command. They are mapped by the DFHSMDDS DSECT.

Table 174. Storage manager: Domain subpools statistics

DFHSTUP name	Field name	Description
Subpool Name	SMDSPN	The unique 8-character name of the domain subpool. The values of the domain subpool field are described in CICS subpools in Improving performance. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	SMDETYPE	The assembler DSECT field name indicates whether all the elements in the subpool are fixed length or variable length. • X'01' fixed • X'02' variable For further information about subpool elements, see CICS subpools in Improving performance. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	SMDFLEN	The length of each subpool element (applicable to fixed length subpools only). For further information about subpool elements, see CICS subpools in Improving performance. 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 the storage manager maintains an element chain for the subpool with the addresses and lengths of each element. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	SMDBNDRY	The boundary on which each element is aligned. This is a power of 2 in the range 8 through 4096 bytes. This field does not apply to 64-bit (above-the-bar) storage. Reset characteristic: Not reset

Table 174. Storage manager: Domain subpools statistics (continued)

DFHSTUP name	Field name	Description
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 16 MB (below the line). • SMDABOVE (X'02') above 16 MB but below 2 GB (above the line). • SMDABOVEBAR (X'03') above the bar. Reset characteristic: Not reset
Location	SMDDSANAME	The name of the DSA that the domain subpool is allocated from. Values can be CDSA, SDSA, RDSA, ECDSA, ESDSA, ERDSA, ETDSA, GCDSA, or GSDSA. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	SMDDSAINDEX	A unique identifier for the dynamic storage area that this subpool is allocated from. Values can be as follows: • 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 RDSA. • SMDECDSA (X'09') indicating that the subpool storage is obtained from the ECDSA. • SMDESDSA (X'0B') indicating that the subpool storage is obtained from the ESDSA. • SMDERDSA (X'0C') indicating that the subpool storage is obtained from the ERDSA. • SMDETDSA (X'0D') indicating that the subpool storage is obtained from the ETDSA. • SMDGCDSA (X'11') indicating that the subpool storage is obtained from the GCDSA. • SMDGSDSA (X'13') indicating that the subpool storage is obtained from the GSDSA. Reset characteristic: Not reset
Access	SMDACCESS	The type of access of the subpool. Values are CICS, USER, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the RDSA and ERDSA. • SMDCICS (X'01') access is CICS key. • SMDUSER (X'02') access is USER key. • SMDREADONLY (X'03') is read-only protection. • SMDTRUSTED (X'04') access is CICS key. Reset characteristic: Not reset

Table 174. Storage manager: Domain subpools statistics (continued)

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMDIFREE	The size of the initial free area for the subpool (which might be zero). For further information about the initial free area, see "Defining programs as resident, nonresident, or transient" on page 140. This value is expressed in bytes. Reset characteristic: Not reset
Getmain Requests	SMDGMREQ	The total number of GETMAIN requests for the subpool. Reset characteristic: Reset to zero
Freemain Requests	SMDFMREQ	The total number of FREEMAIN requests for the subpool. Reset characteristic: Reset to zero
Current Elements	SMDCELEM	The current number of storage elements in the subpool. Reset characteristic: Not reset
Current Elem Stg	SMDCES	The sum of the lengths of all the elements in the subpool, expressed in bytes. Reset characteristic: Not reset
Current Page Stg	SMDCPS	The space taken by all the pages allocated to the subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Reset characteristic: Not reset
Peak Page Stg	SMDHWMPS	The peak page storage allocated to support the storage requirements of this subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Reset characteristic: Reset to current value

Related information:

EXTRACT STATISTICS in Reference > System programming

Storage manager: Global statistics

You can retrieve storage manager global statistics by using the **EXEC CICS EXTRACT STATISTICS STORAGE** system command. They are mapped by the DFHSMSDS DSECT.

These statistics are collected for each dynamic storage area (DSA). .

Table 175. Storage manager: Global statistics

DFHSTUP name	Field name	Description
Storage protection	SMSSTGPROT	Whether storage protection is active: • X'01' active • X'00' not active Reset characteristic: Not reset
Transaction isolation	SMSTRANISO	Whether transaction isolation is active: • X'01' active • X'00' not active Reset characteristic: Not reset
Reentrant programs	SMSRENTPGM	Whether write protection for reentrant programs is enabled: • X'01' PROTECT - RDSA and ERDSA are obtained from key 0 storage. • X'00' NOPROTECT - RDSA and ERDSA are obtained from key 8 storage. Reset characteristic: Not reset
Current DSA limit	SMSDSALIMIT	The current limit of the CICS dynamic storage areas, as defined by the **DSALIM** system initialization parameter. Reset characteristic: Not reset
Current DSA total	SMSDSATOTAL	The total amount of storage currently allocated to the DSAs below 16 MB (below the line). This value might be smaller or larger than "Current DSA limit". Reset characteristic: Not reset
Peak DSA total	SMSHWMDSATOTAL	The peak amount of storage allocated to the DSAs below 16 MB (below the line). This value might be smaller or larger than "Current DSA limit". Reset characteristic: Reset to current value
Current EDSA limit	SMSEDSALIMIT	The current limit of the CICS extended dynamic storage areas, as defined by the **EDSALIM** system initialization parameter. Reset characteristic: Not reset
Current EDSA total	SMSEDSATOTAL	The total amount of storage currently allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than "Current EDSA limit". Reset characteristic: Not reset
Peak EDSA total	SMSHWMESDATOTAL	The peak amount of storage allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than "Current EDSA limit". Reset characteristic: Reset to current value
MEMLIMIT size	SMSMEMLIMIT	The value of the z/OS **MEMLIMIT** parameter, which limits the amount of 64-bit storage for the CICS region. This value can be in megabytes, gigabytes, terabytes, petabytes, or exabytes, depending on size. A value of NOLIMIT indicates that no upper limit is imposed. Reset characteristic: Not reset

Table 175. Storage manager: Global statistics (continued)

DFHSTUP name	Field name	Description
MEMLIMIT set by	SMSMEMLIMITSRC	The source of the **MEMLIMIT** value: SMFPRM indicates that **MEMLIMIT** is set by SYS1.PARMLIB(SMFPRMxx). JCL indicates that **MEMLIMIT** is set by JCL. REGION indicates that **MEMLIMIT** is set to NOLIMIT because REGION=0M is specified in JCL. IEFUSI indicates that **MEMLIMIT** is set by the z/OS installation exit IEFUSI. Reset characteristic: Not reset
GETSTOR request size	SMSGETSTORSIZE	The GETSTOR request size. Reset characteristic: Not reset
Current Address Space active	SMSASACTIVE	The current address space available above the bar. Reset characteristic: Not reset
Peak Address Space active	SMSHWMASACTIVE	The peak amount of address space available above the bar. Reset characteristic: Reset to current value
Current GDSA active	SMSGDSAACTIVE	The current storage in use above the bar. Reset characteristic: Not reset
Peak GDSA active	SMSHWMGDSAACTIVE	The peak amount of storage in use above the bar. Reset characteristic: Reset to current value
MVS storage request waits	SMSMVSSTGREQWAITS	The total number of MVS storage requests that have waited for MVS storage above 16 MB. Reset characteristic: Reset to zero
Total time waiting for MVS storage	SMSTIMEWAITMVS	The total time that MVS storage requests have spent waiting for MVS storage above 16 MB. Reset characteristic: Reset to zero
Bytes Allocated to Private Memory Objects	SMSLVABYTES	The number of bytes allocated from large virtual memory in private memory objects. [1] Reset characteristic: Not reset
Bytes Hidden within Private Memory Objects	SMSLVHBYTES	The number of bytes hidden in large virtual memory private memory objects. [1] Reset characteristic: Not reset
Peak Bytes Usable within Private Memory Objects	SMSLVGBYTES	The high-water mark of usable bytes in large virtual memory private memory objects. [1] Reset characteristic: Not reset
Number of Private Memory Objects	SMSLVNMEMOBJ	The number of private memory objects allocated. [1] Reset characteristic: Not reset
Auxiliary Slots backing Private Memory Objects	SMSHVAUXSLOTS	The number of auxiliary storage slots that are used to back 64-bit private memory objects. [1] Reset characteristic: Not reset
HWM Auxiliary Slots backing Private Memory Objects	SMSHVGAUXSLOTS	The high-water mark of auxiliary storage slots that are used to back 64-bit private memory objects. [1] Reset characteristic: Not reset

Table 175. Storage manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Real Frames backing Private Memory Objects	SMSHVPAGESINREAL	The number of real storage frames that are used to back 64-bit private memory objects. [1] Reset characteristic: Not reset
HWM Real Frames backing Private Memory Objects	SMSHVGPAGESINREAL	The high-water mark for the number of real storage frames that are used to back 64-bit private memory objects. [1] Reset characteristic: Not reset
Number of Large Memory Objects Allocated	SMSLARGEMEMOBJ	The number of large memory objects allocated by this address space. [1] Reset characteristic: Not reset
Number of Large Pages backed in Real Storage	SMSLARGEPAGESINREAL	The number of large pages (1 MB pages) backed in real storage owned by this address space. [1] Reset characteristic: Not reset
Shared Bytes from Large Memory Objects	SMSLVSHRBYTES	The number of shared bytes allocated from high virtual memory. [1] Reset characteristic: Not reset
Peak Shared Bytes within Large Memory Objects	SMSLVSHRGBYTES	The high-water mark for the number of shared bytes in large virtual memory objects. [1] Reset characteristic: Not reset
Number of Shared Memory Objects	SMSLVSHRNMEMOBJ	The number of shared memory objects allocated.[1] Reset characteristic: Not reset
Number of FROMGUARD Failures	SMSFROMGUARDFAIL	The number of times that a request for 64-bit storage has failed, where the request uses the z/OS IARV64 macro with the REQUEST=CHANGEGUARD, CONVERT=FROMGUARD parameters. [1] Reset characteristic: Reset to zero
FROMGUARD Failure size	SMSFROMGUARDFAILSIZE	The size of the largest request for 64-bit storage that has failed, in bytes, where the request uses the z/OS IARV64 macro with the REQUEST=CHANGEGUARD, CONVERT=FROMGUARD parameters. [1] Reset characteristic: Reset to zero
Current GDSA allocated	SMSGDSAALLOC	The total amount of storage currently allocated to the DSAs above the bar. Reset characteristic: Not reset
Peak GDSA allocated	SMSHWMGDSAALLOC	The peak amount of storage allocated to the DSAs above the bar. Reset characteristic: Reset to current value

Note:

1. For more information about the memory that this statistic refers to, see Using the 64-bit Address Space in the z/OS MVS Programming: Extended Addressability Guide.

Related information:

EXTRACT STATISTICS in Reference > System programming

Storage manager: Subspace statistics

You can retrieve storage manager subspace statistics by using the **EXEC CICS EXTRACT STATISTICS STORAGE** system command. They are mapped by the DFHSMSDS DSECT.

These statistics are collected for each DSA.

Table 176. Storage manager: Subspace statistics

DFHSTUP name	Field name	Description
Current unique subspace users	SMSUSSCUR	Current number of unique subspace users. Number of tasks currently allocated a unique subspace. **Reset characteristic:** Not reset.
Total unique subspace users	SMSUSSCUM	Total number of tasks that have been allocated a unique subspace. **Reset characteristic:** Reset to zero.
Peak unique subspace users	SMSUSSHWM	The peak number of tasks concurrently allocated a unique subspace. **Reset characteristic:** Reset to current.
Current common subspace users	SMSCSSCUR	Number of tasks currently allocated to the common subspace **Reset characteristic:** Not reset.
Total common subspace users	SMSCSSCUM	Total number of tasks allocated to the common subspace **Reset characteristic:** Reset to zero.
Peak common subspace users	SMSCSSHWM	The peak number of tasks concurrently allocated to the common subspace. **Reset characteristic:** Reset to current.

Related information:

EXTRACT STATISTICS in Reference > System programming

Storage manager: Dynamic storage areas statistics

You can retrieve storage manager dynamic storage area statistics by using the **EXEC CICS EXTRACT STATISTICS STORAGE** system command. They are mapped by the DFHSMSDS DSECT.

The dynamic storage areas statistics are collected for each DSA.

Note: All the fields, except NOT IN THE DFHSETUP REPORT, are mapped by the SMSBODY DSECT within the DFHSMSDS DSECT. The SMSBODY DSECT is repeated for each DSA in the CICS region (SMSNPAGP).

Table 177. Storage manager: Dynamic storage areas statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMSNPAGP	The number of DSAs in the CICS region. There are 12 DSAs: • CDSA, UDSA, SDSA, RDSA • ECDSA, EUDSA, ESDSA, ERDSA, ETDSA • GCDSA, GUDSA, GSDSA Reset characteristic: Not reset

Table 177. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Header in DFHSTUP report	SMSDSANAME	Name of the DSA that this record represents. The value can be CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. 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'09'). The page pool is the ECDSA. • SMSEUDSA (X'0A'). The page pool is the EUDSA. • SMSESDSA (X'0B'). The page pool is the ESDSA. • SMSERDSA (X'0C'). The page pool is the ERDSA. • SMSETDSA (X'0D'). The page pool is the ETDSA. • SMSGCDSA (X'11'). The page pool is the GCDSA. • SMSGUDSA (X'12'). The page pool is the GUDSA. • SMSGSDSA (X'13'). The page pool is the GSDSA. Reset characteristic: Not reset
NOT IN THE DFHSTUP REPORT	SMSLOCN	The location of this DSA. The assembler DSECT field name has the following values: • SMSBELOW (X'01') below the 16 MB line • SMSABOVE (X'02') above 16 MB but below 2 GB • SMSABOVEBAR (X'03') above the bar Reset characteristic: Not reset
Current DSA Size	SMSDSASZ	The current size of the DSA. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes. Reset characteristic: Not reset
Peak DSA Size	SMSHWMDSASZ	The peak size of the DSA since the last time that statistics were recorded. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes. Reset characteristic: Reset to current value

Table 177. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Cushion Size	SMSCSIZE	The size of the cushion. The cushion forms part of each DSA and is the amount of storage below which CICS goes short on storage (SOS). For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes. Reset characteristic: Not reset
Free storage (inc. cushion)	SMSFSTG	The amount of free storage in this DSA; that is, the number of free pages multiplied by the page size. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, the page size is 4 KB and this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, the page size is 1 MB and this value is expressed in megabytes. Reset characteristic: Not reset
Percentage free storage		The percentage of the storage that is free. This value is calculated offline by DFHSTUP and is, therefore, not accessible from the **EXEC CICS EXTRACT STATISTICS** command. This field does not apply to the GCDSA, GUDSA, and GSDSA. Reset characteristic: Not reset
Peak free storage	SMSHWMFSTG	The peak amount of free storage in this DSA since the last time that statistics were recorded. Free storage is the number of free pages multiplied by the page size. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, the page size is 4 KB and this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, the page size is 1 MB and this value is expressed in megabytes. Reset characteristic: Reset to current value
Lowest free storage	SMSLWMFSTG	The smallest amount of free storage in this DSA since the last time that statistics were recorded. Free storage is the number of free pages multiplied by the page size. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, the page size is 4 KB and this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, the page size is 1 MB and this value is expressed in megabytes. Reset characteristic: Reset to current value

Table 177. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Largest free area	SMSLFA	The length of the largest contiguous free area in this DSA. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes. For an indication of the storage fragmentation in this DSA, compare this value with "Free storage" (SMSFSTG) in the DSA. If the ratio is large, this DSA is fragmented. Reset characteristic: Not reset
Getmain Requests	SMSGMREQ	The number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. Reset characteristic: Reset to zero
Freemain Requests	SMSFMREQ	The number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. Reset characteristic: Reset to zero
Add Subpool Requests	SMSASR	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. Reset characteristic: Reset to zero
Delete Subpool Requests	SMSDSR	The number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. Reset characteristic: Reset to zero
Current no of Subpools	SMSCSUBP	The current number of subpools (domain and task) in the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA. Reset characteristic: Not reset
Times no storage returned	SMSCRISS	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Reset characteristic: Reset to zero
Times request suspended	SMSUCSS	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at that moment. Reset characteristic: Reset to zero

Table 177. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Current suspended	SMSCSS	The number of GETMAIN requests that are currently suspended for storage. Reset characteristic: Not reset
Peak requests suspended	SMSHWMSS	The peak number of GETMAIN requests that were suspended for storage. Reset characteristic: Reset to current value
Purged while waiting	SMSPWWS	The number of requests that were purged while suspended for storage. Reset characteristic: Reset to zero
Times cushion released	SMSCREL	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 and there are no more free extents available to increase the size of this DSA. Reset characteristic: Reset to zero
Times went short on storage	SMSSOS	The number of times CICS went SOS in this DSA, where SOS means that the cushion is currently in use, or at least one task is suspended for storage, or both. This field applies to CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, and GSDSA. Reset characteristic: Reset to zero
Total time SOS	SMSTSOS	The accumulated time that CICS has been SOS in this DSA. The DFHSTUP report expresses this time as *days:hours:minutes: seconds.decimals*. The DSECT field contains the time as a store clock (STCK) value. Reset characteristic: Reset to zero
Storage violations	SMSSV	The number of storage violations recorded in the DSA. Reset characteristic: Reset to zero

Table 177. Storage manager: Dynamic storage areas statistics (continued)

DFHSTUP name	Field name	Description
Access	SMSACCESS	The type of access of the DSA. Values are CICS, USER, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the RDSA or ERDSA. • SMSCICS (X'01') access is CICS key. • SMSUSER (X'02') access is USER key. • SMSREADONLY (X'03') is read-only protection. • SMSTRUSTED (X'04') access is CICS key. Reset characteristic: Not reset
Current extents	SMSEXTS	The number of extents currently allocated to this DSA. Reset characteristic: Not reset
Extents added	SMSEXTSA	The number of extents added to the DSA since the last time statistics were recorded. Reset characteristic: Reset to zero
Extents released	SMSEXTSR	The number of extents that were released from the DSA since the last time statistics were recorded. Reset characteristic: Reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Storage manager: Task subpools statistics

Task subpools statistics are collected for each dynamic storage area (DSA). They are mapped by the DFHSMTDS DSECT.

These statistics are produced for offline processing (written to SMF). They cannot be accessed online by using the **EXTRACT STATISTICS** command.

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 DSAs (CDSA, UDSA, ECDSA, EUDSA, GCDSA, and GUDSA). If further granularity of task storage usage is required, use the performance class data of the CICS monitoring facility.

Apart from the SMTNTASK field, the fields in the following table are mapped by the SMTBODY DSECT in the DFHSMTDS DSECT. The SMTBODY DSECT is repeated for each task subpool in the CICS region (SMTNTASK).

Table 178. Storage manager: Task subpools statistics

DFHSTUP name	Field name	Description
NOT IN THE DFHSTUP REPORT	SMTNTASK	The number of task subpools in the CICS region. Reset characteristic: not reset
DSA Name	SMTDSANAME	The name of the dynamic storage area from which this task storage has been allocated. Values can be CDSA, UDSA, ECDSA, EUDSA, GCDSA, and GUDSA. 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'09'), indicating that the task storage is obtained from the ECDSA • SMTEUDSA (X'0A'), indicating that the task storage is obtained from the EUDSA • SMTGCDSA (X'11'), indicating that the task storage is obtained from the GCDSA • SMTGUDSA (X'12'), indicating that the task storage is obtained from the GUDSA Reset characteristic: not reset
NOT IN THE DFHSTUP REPORT	SMTLOCN	Indicates the storage location of the DSA: • SMTBELOW (X'01') below the 16 MB line • SMTABOVE (X'02') above 16 MB but below 2 GB (above the line) • SMTABOVEBAR (X'03') above the bar Reset characteristic: not reset
Access	SMTACCESS	The type of access of the subpool. Access type can be CICS (key 8) or USER (key 9). • SMTCICS (X'01') access is CICS key • SMTUSER (X'02') access is USER key Reset characteristic: not reset
Getmain Requests	SMTGMREQ	The total number of task subpool GETMAIN requests from this dynamic storage area. Reset characteristic: reset to zero

Table 178. Storage manager: Task subpools statistics (continued)

DFHSTUP name	Field name	Description
Freemain Requests	SMTFMREQ	The total number of task subpool FREEMAIN requests from this dynamic storage area. Reset characteristic: reset to zero
Current Elements	SMTCNE	The number of elements in all the task subpools in this dynamic storage area. Reset characteristic: not reset
Current Elem Stg	SMTCES	The sum of the storage occupied by all elements in task subpools in this dynamic storage area, expressed in bytes. Reset characteristic: not reset
Current Page Stg	SMTCPS	The sum of the storage in all pages allocated to task subpools in this dynamic storage area. This value is expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Reset characteristic: not reset
Peak Page Stg	SMTHWMPS	The peak page storage allocated to support task storage activity in this dynamic storage area. This value is expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Reset characteristic: reset to current value

Storage manager: Summary domain subpools statistics

Shows summary information and statistics about domain subpools.

Summary statistics are not available online.

Table 179. Storage manager: Summary domain subpools statistics

DFHSTUP name	Description
Subpool Name	The unique 8-character name of the domain subpool. The values of the domain subpool field are described in CICS subpools in Improving performance.
Location	The name of the DSA that the domain subpool is allocated from. Values can be CDSA, SDSA, RDSA, ECDSA, ESDSA, ERDSA, ETDSA, GCDSA, or GSDSA.

Table 179. Storage manager: Summary domain subpools statistics (continued)

DFHSTUP name	Description
Access	The type of access of the subpool. Values are CICS, USER, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the RDSA and ERDSA. • SMDCICS (X'01') access is CICS key. • SMDUSER (X'02') access is USER key. • SMDREADONLY (X'03') is read-only protection. • SMDTRUSTED (X'04') access is CICS key.
Getmain Requests	The total number of GETMAIN requests for the subpool.
Freemain Requests	The total number of FREEMAIN requests for the subpool.
Peak Elements	The peak number of storage elements in the subpool.
Peak Elem Stg	The peak amount of element storage in the subpool, expressed in bytes.
Peak Page Stg	The peak page storage allocated to support the storage requirements of this subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage).

Storage manager: Summary global statistics

Storage manager summary global statistics are not available online.

Table 180. Storage manager: Summary global statistics

DFHSTUP name	Description
Storage protection	Whether storage protection is active: • X'01' active • X'00' not active
Transaction isolation	Whether transaction isolation is active: • X'01' active • X'00' not active
Reentrant programs	Whether write protection for reentrant programs is enabled: • X'01' PROTECT - RDSA and ERDSA are obtained from key 0 storage. • X'00' NOPROTECT - RDSA and ERDSA are obtained from key 8 storage.
Current DSA limit	The current limit of the CICS dynamic storage areas, as defined by the **DSALIM** system initialization parameter.
Current DSA total	The total amount of storage currently allocated to the DSAs below 16 MB (below the line). This value might be smaller or larger than “Current DSA limit”.

Table 180. Storage manager: Summary global statistics (continued)

DFHSTUP name	Description
Peak DSA total	The peak amount of storage allocated to the DSAs below 16 MB (below the line). This value might be smaller or larger than "Current DSA limit".
Current EDSA limit	The current limit of the CICS extended dynamic storage areas, as defined by the **EDSALIM** system initialization parameter.
Current EDSA total	The total amount of storage currently allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than "Current EDSA limit".
Peak EDSA total	The peak amount of storage allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than "Current EDSA limit".
MEMLIMIT size	The value of the z/OS **MEMLIMIT** parameter, which limits the amount of 64-bit storage for the CICS region. This value can be in megabytes, gigabytes, terabytes, petabytes, or exabytes, depending on size. A value of NOLIMIT indicates that no upper limit is imposed.
MEMLIMIT set by	The source of the **MEMLIMIT** value: SMFPRM indicates that **MEMLIMIT** is set by SYS1.PARMLIB(SMFPRMxx). JCL indicates that **MEMLIMIT** is set by JCL. REGION indicates that **MEMLIMIT** is set to NOLIMIT because REGION=0M is specified in JCL. IEFUSI indicates that **MEMLIMIT** is set by the z/OS installation exit IEFUSI.
Current GDSA allocated	The total amount of storage currently allocated to the DSAs above the bar.
Peak GDSA allocated	The peak amount of storage allocated to the DSAs above the bar.
Current GDSA active	The current storage in use above the bar.
Peak GDSA active	The peak amount of storage in use above the bar.
MVS storage request waits	The total number of MVS storage requests that have waited for MVS storage above 16 MB.
Total time waiting for MVS storage	The total time that MVS storage requests have spent waiting for MVS storage above 16 MB.

Storage manager: Summary subspace statistics

Storage manager summary subspace statistics are not available online.

Table 181. Storage manager: Summary subspace statistics

DFHSTUP name	Description
Total unique subspace users	The total number of tasks that have been allocated a unique subspace.
Peak unique subspace users	The peak number of tasks concurrently allocated a unique subspace.
Total common subspace users	The total number of tasks allocated to the common subspace.

Table 181. Storage manager: Summary subspace statistics (continued)

DFHSTUP name	Description
Peak common subspace users	The peak number of tasks concurrently allocated to the common subspace.

Storage manager: Summary dynamic storage areas statistics

Shows summary information and statistics about dynamic storage areas.

Summary statistics are not available online.

Table 182. Storage manager: Summary dynamic storage areas statistics

DFHSTUP name	Description
Current DSA size	The current size of the DSA. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes.
Peak DSA size	The peak size of the DSA since the last time that statistics were recorded. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes.
Cushion size	The size of the cushion. The cushion forms part of each DSA and is the amount of storage below which CICS goes short on storage (SOS). For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, this value is expressed in megabytes.
Peak free storage	The peak amount of free storage in this DSA since the last time that statistics were recorded. Free storage is the number of free pages multiplied by the page size. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, the page size is 4 KB and this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, the page size is 1 MB and this value is expressed in megabytes.
Lowest free storage	The smallest amount of free storage in this DSA since the last time that statistics were recorded. Free storage is the number of free pages multiplied by the page size. For the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, the page size is 4 KB and this value is expressed in bytes. For the GCDSA, GUDSA, and GSDSA, the page size is 1 MB and this value is expressed in megabytes.
Getmain requests	The number of GETMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA.
Freemain requests	The number of FREEMAIN requests from the CDSA, UDSA, SDSA, RDSA, ECDSA, EUDSA, ESDSA, ERDSA, ETDSA, GCDSA, GUDSA, or GSDSA.
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE.

Table 182. Storage manager: Summary dynamic storage areas statistics (continued)

DFHSTUP name	Description
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at that moment.
Peak requests suspended	The peak number of GETMAIN requests that were suspended for storage.
Purged while waiting	The number of requests that were purged while suspended for storage.
Times cushion released	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 and there are no more free extents available to increase the size of this DSA.
Times went short on storage	The number of times CICS went SOS in this DSA, where SOS means that the cushion is currently in use, or at least one task is suspended for storage, or both.
Total time SOS	The accumulated time that CICS has been SOS in this DSA.
Storage violations	The number of storage violations recorded in the DSA.
Access	The type of access of the DSA. Values are CICS, USER, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the RDSA or ERDSA.
Current extents	The number of extents currently allocated to this DSA.
Extents added	The number of extents added to the DSA since the last time statistics were recorded.
Extents released	The number of extents that were released from the DSA since the last time statistics were recorded.

Storage manager: Summary task subpools statistics

This report shows summary information and statistics about task subpools.

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).

Table 183. Storage manager: Summary task subpools statistics

DFHSTUP name	Description
DSA Name	The name of the dynamic storage area from which this task storage has been allocated. Values can be CDSA, UDSA, ECDSA, EUDSA, GCDSA, and GUDSA.
Access	The type of access of the subpool. Access type can be CICS (key 8) or USER (key 9).
Getmain Requests	The total number of task subpool GETMAIN requests from this dynamic storage area.
Freemain Requests	The total number of task subpool FREEMAIN requests from this dynamic storage area.
Peak Elements	The peak of the current number of elements in all the task subpools in this dynamic storage area.
Peak Elem Stg	The peak of the current amount of storage occupied by all elements in task subpools within this dynamic storage area, expressed in bytes.
Peak Page Stg	The peak page storage allocated to support task storage activity in this dynamic storage area.

Table manager statistics

Table manager: Global statistics

You can retrieve table manager global statistics by using the **EXEC CICS COLLECT STATISTICS TABLEMGR** system command. They are mapped by the DFHA16DS DSECT.

Table 184. Table manager: Global statistics. Apart from the first field, the following fields are mapped by the A16STATS DSECT, which is repeated for each table (A16NTAB).

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
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 Table Name field (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves. Reset characteristic: not reset

Related reference:

COLLECT STATISTICS in Reference > System programming

Table manager: Summary global statistics

Table manager summary global statistics are not available online.

Table 185. Table manager: Summary global statistics

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 Table Name field (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 Table Name field (for example, for scatter tables and directory segments). This does not include storage used by the tables themselves.

TCP/IP global and TCP/IP Service statistics

TCP/IP support is the basis for CICS web support and web services in CICS. Each port on which TCP/IP requests can be received is defined by a TCPIPSERVICE resource definition. The statistics include global statistics and statistics for each TCPIPSERVICE definition.

DFH0STAT reports: See TCP/IP report and TCP/IP services report

Related reference:

"TCP/IP report" on page 892
The TCP/IP report is produced using a combination of **EXEC CICS INQUIRE TCPIP** and **EXEC CICS EXTRACT STATISTICS TCPIP** commands. The statistics data is mapped by the **DFHSOGDS DSECT**.

"TCP/IP services report" on page 895
The TCP/IP services report is produced using a combination of **EXEC CICS INQUIRE TCPIPSERVICE** and **EXEC CICS EXTRACT STATISTICS TCPIPSERVICE** commands. The statistics data is mapped by the **DFHSORDS DSECT**.

TCP/IP: Global statistics

You can retrieve TCP/IP global statistics by using the **EXEC CICS EXTRACT STATISTICS TCPIP** system command. They are mapped by the DFHSOGDS DSECT.

Table 186. TCP/IP: Global statistics

DFHSTUP name	Field name	Description
Current number of inbound sockets	SOG_CURR_INBOUND_SOCKETS	is the current number of inbound sockets. Reset characteristic: not reset

Table 186. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
Peak number of inbound sockets	SOG_PEAK_INBOUND_SOCKETS	is the peak number of inbound sockets. Reset characteristic: reset to current
Current number of non-persistent outbound sockets	SOG_CURR_OUTB_SOCKETS	is the current number of non-persistent outbound sockets. Reset characteristic: not reset
Peak number of non-persistent outbound sockets	SOG_PEAK_OUTB_SOCKETS	is the peak number of non-persistent outbound sockets. Reset characteristic: reset to current
Current number of persistent outbound sockets	SOG_CURR_PERS_OUTB_SOCKETS	is the current number of persistent outbound sockets. Reset characteristic: not reset
Peak number of persistent outbound sockets	SOG_PEAK_PERS_OUTB_SOCKETS	is the peak number of persistent outbound sockets. Reset characteristic: reset to current
Total number of inbound sockets created	SOG_INB_SOCKETS_CREATED	is the total number of inbound sockets created. Reset characteristic: reset to zero
Total number of outbound sockets created	SOG_OUTB_SOCKETS_CREATED	is the total number of outbound sockets created. Reset characteristic: reset to zero
Total number of outbound sockets closed	SOG_OUTB_SOCKETS_CLOSED	is the total number of outbound sockets closed. Reset characteristic: reset to zero
Total number of inbound and outbound sockets created	SOG_INB_SOCKETS_CREATED + SOG_OUTB_SOCKETS_CREATED	is the total number of inbound and outbound sockets created. Reset characteristic: reset to zero

Table 186. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
SSLCACHE setting	SOG_SSLCACHE	reports whether SSL caching is taking place locally within a CICS region, or across a sysplex. Reset characteristic: not reset
Current MAXSOCKETS limit	SOG_MAXSOCKETS_LIMIT	is the maximum number of IP sockets that can be managed by the CICS sockets domain. Reset characteristic: not reset
Number of times the MAXSOCKETS limit was reached	SOG_TIMES_AT_MAX_SOCKETS	is the number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached. Reset characteristic: reset to zero
Number of create socket requests delayed by MAXSOCKETS limit	SOG_DELAYED_AT_MAX_SOCKETS	is the number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit. Reset characteristic: reset to zero
Total MAXSOCKETS delay time	SOG_QTIME_AT_MAX_SOCKETS	is the total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit. Reset characteristic: reset to zero
Number of create sockets requests timed out at MAXSOCKETS	SOG_TIMEDOUT_AT_MAX_SOCKETS	is the number of create socket requests that were timed out while delayed because the system had reached the MAXSOCKETS limit. Reset characteristic: reset to zero
Current create socket requests delayed by MAXSOCKETS limit	SOG_CURR_DELAYED_AT_MAX	is the current number of create socket requests delayed because the system is at the MAXSOCKETS limit. Reset characteristic: not reset
Peak create socket requests delayed at MAXSOCKETS	SOG_PEAK_DELAYED_AT_MAX	is the peak number of create socket requests delayed becuase the system is at the MAXSOCKETS limit. Reset characteristic: reset to current

Table 186. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
Current MAXSOCKETS delay time	SOG_CURRENT_QTIME_AT_MAX	is the current total delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. Reset characteristic: not reset
Performance tuning for HTTP connections	SOG_SOTUNING	indicates whether performance tuning for HTTP connections will take place. Reset characteristic: not reset
Socket listener has paused listening for HTTP connections	SOG_PAUSING_HTTP_LISTENING	indicates whether the listener has paused listening for HTTP connection requests because the number of tasks in the region has reached the limit for accepting new HTTP connection requests Reset characteristic: not reset
Number of times socket listener notified at task accept limit	SOG_TIMES_AT_ACCEPT_LIMIT	is the number of times the listener has been notified that the number of tasks in the region has reached the limit for accepting new HTTP connection requests. Reset characteristic: reset to zero
Last time socket listener paused listening for HTTP connections	SOG_TIME_LAST_PAUSED_HTTP_LISTENING	is the last time the socket listener paused listening for HTTP connection requests because the number of tasks in the region had reached the limit for accepting new HTTP connection requests. The DFHSTUP report expresses this time as day/month/year hours:minutes:seconds:decimals; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that the listener has never paused listening for HTTP connection requests since the statistics were last reset. Reset characteristic: reset to zero
Region stopping HTTP connection persistence	SOG_STOPPING_PERSISTENCE	indicates whether the region is stopping HTTP connection persistence because the number of tasks in the region has exceeded the limit. Reset characteristic: not reset

Table 186. TCP/IP: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of times region stopped HTTP connection persistence	SOG_TIMES_STOPPED_PERSISTENT	is the number of times the region took action to stop HTTP connection persistence because the number of tasks in the region has exceeded the limit. Reset characteristic: reset to zero
Last time stopped HTTP connection persistence	SOG_TIME_LAST_STOPPED_PERSISTENT	is the last time the region took action to stop HTTP connection persistence because the number of tasks in the region has exceeded the limit. The DFHSTUP report expresses this time as day/month/year hours:minutes:seconds:decimals; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that HTTP connection persistence has not been stopped since the statistics were last reset. Reset characteristic: reset to zero
Number of persistent connections made non-persistent	SOG_TIMES_MADE_NON_PERSISTENT	is the number of times a persistent HTTP connection was made non-persistent because the number of tasks in the region has exceeded the limit. Reset characteristic: reset to zero
Number of times disconnected an HTTP connection at max uses	SOG_TIMES_CONN_DISC_AT_MAX	is the number of times a persistent HTTP connection was disconnected because the number of uses had exceeded the limit. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

TCP/IP: Summary global statistics

TCP/IP summary global statistics are not available online.

Table 187. TCP/IP: Summary global statistics

DFHSTUP name	Description
Peak number of inbound sockets	is the peak number of inbound sockets.
Peak number of non-persistent outbound sockets	is the peak number of non-persistent outbound sockets.

Table 187. TCP/IP: Summary global statistics (continued)

DFHSTUP name	Description
Peak number of persistent outbound sockets	is the peak number of persistent outbound sockets.
Total number of inbound sockets created	is the total number of inbound sockets created.
Total number of outbound sockets created	is the total number of outbound sockets created.
Total number of outbound sockets closed	is the total number of outbound sockets closed.
Total number of inbound and outbound sockets created	is the total number of inbound and outbound sockets created.
SSLCACHE setting	reports whether SSL caching is taking place locally within a CICS region, or across a sysplex.
MAXSOCKETS limit	is the maximum number of IP sockets that can be managed by the CICS sockets domain.
Times the MAXSOCKETS limit was reached	is the number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached.
Total number of create socket requests timed out at MAXSOCKETS	is the total number of create socket requests that were timed out while delayed because the system had reached the MAXSOCKETS limit.
Peak number of create socket requests delayed at MAXSOCKETS	is the peak number of create socket requests delayed becuase the system was at the MAXSOCKETS limit.
Total number of create socket requests delayed at MAXSOCKETS	is the total number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit.
Total MAXSOCKETS delay time	is the total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit.
Average MAXSOCKETS delay time	is the average time that create socket requests were delayed because the system had reached the MAXSOCKETS limit.

TCP/IP services: Resource statistics

You can retrieve TCP/IP services resource statistics by using the **EXEC CICS EXTRACT STATISTICS TCPIPSERVICE** system command. They are mapped by the TCPIPSERVICE and the DFHSORDS DSECTs.

Table 188. TCP/IP Services: resource statistics

DFHSTUP name	Field name	Description
TCPIPSERVICE Name	SOR_SERVICE_NAME	The name of the TCP/IP service Reset characteristic: not reset
TCPIPSERVICE Open Date/Time	SOR_OPEN_LOCAL	The date and time on which this TCP/IP service was opened. If this field is not set, SOR_OPEN_LOCAL contains the hexadecimal value X'0000000000000000', shown in the report as "CLOSED". If the field is set, it contains a date expressed in *mm/dd/yyyy* format. This field contains a valid date if the following statements apply: • The TCP/IP service is open at the time the statistics are taken. • The statistics request is unsolicited because the TCP/IP service is closed. Reset characteristic: not reset
TCPIPSERVICE Close Date/Time	SOR_CLOSE_LOCAL	The date and time on which this TCP/IP service was closed. If this field is not set, SOR_CLOSE_LOCAL contains the hexadecimal value X'0000000000000000', 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
TCPIPSERVICE Protocol	SOR_PROTOCOL	The protocol defined for this TCP/IP service. This protocol can be “ECI”, “HTTP”, “IIOP”, “IPIC", "USER", or blank (which means HTTP). Reset characteristic: not reset
TCPIPSERVICE Port	SOR_PORT_NUMBER	The port number being used for this TCP/IP service. Reset characteristic: not reset
TCPIPSERVICE Host	SOR_HOSTNAME	The hostname or IPv4 or IPv6 address of the remote system. Reset characteristic: not reset

Table 188. TCP/IP Services: resource statistics (continued)

DFHSTUP name	Field name	Description
TCPIPSERVICE IP Family	SOR_IP_FAMILY	The address format of the address returned in IP Resolved Address. Reset characteristic: not reset
TCPIPSERVICE IP Resolved Address	SOR_IP_ADDRESS	The IPv4 or IPv6 resolved address of the host. Reset characteristic: not reset
TCPIPSERVICE Transaction ID	SOR_TCPIPS_TRANID	The ID of the CICS transaction attached to process new requests received for this service. Reset characteristic: not reset
TCPIPSERVICE Backlog	SOR_BACKLOG	The port backlog for this TCP/IP service. Reset characteristic: not reset
TCPIPSERVICE URM	SOR_TCPIPS_URM	The name of a user-replaceable program to be called by this service. Reset characteristic: not reset
TCPIPSERVICE Maxdata	SOR_MAXDATA_LENGTH	The maximum length of data that can be received on this TCP/IP service. Reset characteristic: not reset
TCPIPSERVICE SSL Type	SOR_SSL_SUPPORT	The level of SSL support defined for this TCP/IP service. Reset characteristic: not reset
TCPIPSERVICE Authenticate	SOR_AUTHENTICATE	The authentication and identification scheme specified for this TCP/IP service. Reset characteristic: not reset
TCPIPSERVICE Privacy	SOR_PRIVACY	The level of SSL encryption support that applies to this TCP/IP service. Reset characteristic: not reset

Table 188. TCP/IP Services: resource statistics (continued)

DFHSTUP name	Field name	Description
TCPIPSERVICE Attachsec	SOR_ATTACHSEC	The level of attach-time security required for this TCP/IP service. Reset characteristic: not reset
Current Connections	SOR_CURRENT_CONNS	The current number of connections for the TCP/IP service. Reset characteristic: reset to zero
Peak Connections	SOR_PEAK_CONNS	The peak number of connections for the TCP/IP service. Reset characteristic: reset to zero
Transactions Attached	SOR_TRANS_ATTACHED	The number of transactions attached by this TCP/IP Service. Reset characteristic: reset to zero
Send requests	SOR_SENDS	The number of send requests issued for the TCP/IP Service. Reset characteristic: reset to zero
Total Bytes Sent	SOR_BYTES_SENT	The number of bytes sent for the TCP/IP service. Reset characteristic: reset to zero
Receive requests	SOR_RECEIVES	The number of receive requests issued for the TCP/IP Service. Reset characteristic: reset to zero
Total Bytes Received	SOR_BYTES_RECEIVED	The number of bytes received for the TCP/IP service. Reset characteristic: reset to zero
Maximum Persistent Connections	SOR_TCPIPS_MAX_PERSIST	The maximum number of persistent connections from Web clients that the CICS region accepts at any one time. Reset characteristic: not reset

Table 188. TCP/IP Services: resource statistics (continued)

DFHSTUP name	Field name	Description
Non-Persistent Connections	SOR_TCPIPS_NON_PERSIST	The number of connections where CICS did not allow the Web client to have a persistent connection. Reset characteristic: reset to zero
Not in DFHSTUP report	SOR_SERVICE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	SOR_SERVICE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource

signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

TCP/IP services: Summary resource statistics

A summary listing of resource statistics for a TCPIPSERVICE resource.

Summary statistics are not available online.

Table 189. TCP/IP services: summary resource statistics

DFHSTUP name	Description
TCPIPSERVICE Name	The name of the TCPIPSERVICE resource.
TCPIPSERVICE Protocol	The protocol defined for this TCPIPSERVICE resource. This can be ECI, HTTP, IPIC, USER, or blank (which means HTTP).
TCPIPSERVICE Port	The port number being used for this TCPIPSERVICE resource.
TCPIPSERVICE Host	The hostname, IPv4 or IPv6 address of the remote system.
TCPIPSERVICE IP Family	The address format of the address returned in IP Address.
TCPIPSERVICE IP Address	The IPv4 or IPv6 resolved address of the host.
TCPIPSERVICE Transaction ID	The ID of the CICS transaction attached to process new requests received for this service.
TCPIPSERVICE Backlog	The port backlog defined for this TCP/IP service.
TCPIPSERVICE URM	The name of a user-replaceable program to be called by this service.
TCPIPSERVICE Maxdata	The maximum length of data that can be received on this TCP/IP service.
TCPIPSERVICE SSL Type	The level of SSL support defined for this TCP/IP service.
TCPIPSERVICE Authenticate	The authentication and identification scheme specified for this TCP/IP service.

Table 189. TCP/IP services: summary resource statistics (continued)

DFHSTUP name	Description
TCPIPSERVICE Privacy	The level of SSL encryption support that applies to this TCP/IP service.
TCPIPSERVICE Attachsec	The level of attach-time security required for this TCP/IP service.
Peak Connections	The peak number of connections for the TCP/IP Service.
Transactions Attached	The total number of transactions attached for the TCP/IP Service.
Send requests	The total number of send requests issued for the TCP/IP Service.
Total Bytes Sent	The total number of bytes sent for the TCP/IP Service.
Receive requests	The total number of receive requests issued for the TCP/IP Service.
Maximum Persistent Connections	The maximum number of persistent connections from web clients that the CICS region accepts at any one time.
Non-Persistent Connections	The number of connections where CICS did not allow the web client to have a persistent connection.

Temporary storage statistics

Temporary storage statistics are produced for the data that is written into a temporary storage queue.

For more information about how to use these statistics, see Chapter 17, "CICS temporary storage: Performance and tuning," on page 241.

Related concepts:

"Interpreting temporary storage statistics" on page 713
If a data item is written to temporary storage (using WRITEQ TS), a temporary storage queue is built and temporary storage statistics are produced.

Related reference:

"Temporary Storage report" on page 897
The Temporary Storage report is produced using the **EXEC CICS EXTRACT STATISTICS TSQUEUE** command. The statistics data is mapped by the **DFHTSGDS DSECT**.

"Temporary Storage Main — Storage Subpools report" on page 901
The Temporary Storage Main — Storage Subpools report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command. The statistics data is mapped

by the **DFHSMDDS**.

"Temporary Storage Queues report" on page 902
The Temporary Storage Queues report is produced using the **EXEC CICS INQUIRE TSQUEUE** command.

"Tsqueue Totals report" on page 908
The Tsqueue Totals report shows totals that are calculated from data gathered using the **EXEC CICS INQUIRE TSQUEUE** command.

"Temporary Storage Queues by Shared TS Pool report" on page 903
The Temporary Storage Queues by Shared TS Pool report shows temporary storage queues that are in shared TS Pools on the TS Pool servers. These temporary storage queues might or might not currently be in the address space of your system. If they are not in the address space of your system, they are not shown on the other temporary storage queue reports.

"Temporary Storage Models report" on page 902
The Temporary Storage Models report is produced using the **EXEC CICS INQUIRE TSMODEL** command.

Interpreting temporary storage statistics

If a data item is written to temporary storage (using WRITEQ TS), a temporary storage queue is built and temporary storage statistics are produced.

The following statistics might require the actions described:

Writes more than control interval
The number of writes of records whose length was greater than the control interval (CI) size of the TS data set. Use this value 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.

Times aux. storage exhausted
The number of situations where one or more transactions might have been suspended because of a NOSPACE condition, or might have been forced to abend (by using a HANDLE CONDITION NOSPACE command, or using RESP on the WRITEQ TS command, or WRITEQ TS NOSUSPEND command). If this item appears in the statistics, increase the size of the temporary storage data set.

Buffer writes
The number of WRITEs to the temporary storage data set. This includes both WRITEs required for recovery and WRITEs required when the buffer is needed to accommodate another control interval (CI). To minimize input/output activity caused by the second situation, increase buffer allocation. Use the system initialization parameter, TS=(*b*,*s*), where *b* is the number of buffers and *s* is the number of strings.

Peak number of strings in use
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 SIT value to approach this number.

Times string wait occurred
The number of input/output requests that were queued because no strings were available. If this value is not zero, consider increasing the number of strings. For details about adjusting the size of the TS data set and the number of strings and buffers, see Storage calculations for temporary storage data sharing in the *CICS System Definition Guide*.

Temporary storage: Global statistics

You can retrieve temporary storage global statistics by using the **EXEC CICS EXTRACT STATISTICS TSQUEUE** system command. They are mapped by the DFHTSGDS DSECT.

Table 190. Temporary storage: Global statistics

DFHSTUP name	Field name	Description
Put/Putq main storage requests	TSGSTA5F	The number of records that application programs wrote to main temporary storage. Reset characteristic: reset to zero
Get/Getq main storage requests	TSGNMG	The number of records that application programs obtained from main temporary storage. Reset characteristic: reset to zero
Current TSMAINLIMIT setting	TSGTSMLM	The current limit for the amount of storage that CICS makes available for data in main temporary storage. This amount is expressed in bytes. Reset characteristic: not reset
Times at TSMAINLIMIT	TSGTSLHT	The number of times that main temporary storage use attempted to exceed the limit for the amount of storage allowed for data. Reset characteristic: reset to zero
Current storage used for TSMAINLIMIT	TSGTSMUS	The amount of storage that is currently in use for data in main temporary storage. This amount is expressed in bytes. Reset characteristic: not reset
Peak storage used for TSMAINLIMIT	TSGTSMAX	The peak amount of storage that was used for data in main temporary storage. This amount is expressed in bytes. Reset characteristic: reset to current value
Number of queues auto deleted	TSGTSQDL	The number of temporary storage queues that CICS has deleted automatically by using the clean up task. Reset characteristic: reset to zero
Count of clean up task runs	TSGTSCTR	The number of times that the clean up task, which deletes eligible temporary storage queues automatically, has run. Reset characteristic: reset to zero

Table 190. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Put/Putq auxiliary storage requests	TSGSTA7F	The number of records that application programs wrote to auxiliary temporary storage. Reset characteristic: reset to zero
Get/Getq auxiliary storage requests	TSGNAG	The number of records that application programs obtained from auxiliary temporary storage. Reset characteristic: reset to zero
Peak temporary storage names in use	TSGQNUMH	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	TSGQNUM	The current number of temporary storage queue names in use. Reset characteristic: not reset
Number of entries in longest queue	TSGQINH	The peak number of items in any one temporary storage queue, up to a maximum of 32767. Reset characteristic: reset to zero
Times queues created	TSGSTA3F	The number of times that CICS created individual temporary storage queues. Reset characteristic: reset to zero
Control interval size	TSGCSZ	The size of the VSAM 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 control intervals (CIs) permits more data to be transferred at one time, resulting in less system overhead. Reset characteristic: not reset
Available bytes per control interval	TSGNAVB	The number of bytes available for use in the temporary storage data set control interval. Reset characteristic: not reset
Segments per control interval	TSGSPCI	The number of segments available in each temporary storage data set control interval. Reset characteristic: not reset

Table 190. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Bytes per segment	TSGBPSEG	The number of bytes per segment of the temporary storage data set. Reset characteristic: not reset
Writes more than control interval	TSGSTABF	The 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. Reset characteristic: reset to zero
Longest auxiliary temp storage record	TSGLAR	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	TSGNCI	The number of control intervals (CIs) available for auxiliary temporary 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	TSGNCIAH	The peak number of control intervals (CIs) that contain active data. Reset characteristic: reset to current value
Current control intervals in use	TSGNCIA	The current number of control intervals (CIs) that contain active data. Reset characteristic: not reset
Times aux. storage exhausted	TSGSTA8F	The number of situations where one or more transactions might have been suspended because of a NOSPACE condition, or might have been forced to end abnormally (by using a HANDLE CONDITION NOSPACE command). If statistics are present for this field, increase the size of the temporary storage data set. Reset characteristic: reset to zero
Number of temp. storage compressions	TSGSTA9F	The number of times that the temporary storage buffers were compressed. Reset characteristic: reset to zero

Table 190. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Temporary storage buffers	TSGNBCA	The number of temporary storage buffers specified in the TS= system initialization parameter, or in the overrides. The number of buffers allocated might exceed the number requested. Reset characteristic: not reset
Buffer waits	TSGBWTN	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also 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	TSGBUWTH	The peak number of requests queued because no buffers were available. Reset characteristic: reset to current value
Current users waiting on buffer	TSGBUWT	The current number of requests queued because no buffers are available. Reset characteristic: not reset
Buffer writes	TSGTWTN	The number of WRITEs to the temporary storage data set. This includes both WRITEs required for recovery (see Forced writes for recovery) and WRITEs required when the buffer is needed to accommodate another control interval (CI). To minimize input/output activity caused by the second situation, increase buffer allocation. Reset characteristic: reset to zero
Forced writes for recovery	TSGTWTNR	The subset of the total number of WRITEs caused by recovery being specified for queues. This input/output activity is not affected by buffer allocation. Reset characteristic: reset to zero
Buffer reads	TSGTRDN	The number of times a control interval (CI) must be read from disk. To decrease this activity, increase the buffer allocation. Reset characteristic: reset to zero
Format writes	TSGTWTNF	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 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

Table 190. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Temporary storage strings	TSGNVCA	The number of temporary storage strings specified in the TS= system initialization parameter, or in the overrides. The number of strings allocated might exceed the number requested. Reset characteristic: not reset
Peak number of strings in use	TSGNVCAH	The peak number of concurrent input/output operations. If this is significantly less than the number specified in the system initialization table (SIT), consider reducing the SIT value to approach this number. Reset characteristic: reset to current value
Times string wait occurred	TSGVWTN	The number of input/output requests that were queued because no strings were available. If the number of strings is the same as the number of buffers, this number is zero. If this number is a high percentage (over 30%) of the total number of input/output requests (for this purpose, the sum of TSGTWTN, Buffer writes, and TSGTRDN, Buffer reads), consider increasing the number of strings initially allocated. Reset characteristic: reset to zero
Peak number of users waiting on string	TSGVUWTH	The peak number of input/output 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	TSGVUWT	The current number of input/output requests that are queued because all strings are in use. Reset characteristic: not reset
I/O errors on TS data set	TSGSTAAF	The number of input/output errors that occurred on the temporary storage data set. Normally, this number should be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Reset characteristic: reset to zero
Shared pools defined	TSGSHPDF	The number of unique shared TS queue pools defined to CICS. Reset characteristic: reset to zero
Shared pools currently connected	TSGSHPCN	The number of the shared TS pools that this CICS region is connected to. Reset characteristic: reset to zero

Table 190. Temporary storage: Global statistics (continued)

DFHSTUP name	Field name	Description
Shared read requests	TSGSHRDS	The number of TS READQs from the Shared TS Queue pool of TS queues. Reset characteristic: reset to zero
Shared write requests	TSGSHWTS	The number of TS WRITEQs to the Shared TS Queue pool of TS queues. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

Temporary storage: Summary global statistics

Temporary storage summary global statistics are not available online.

Table 191. Temporary storage: Summary global statistics

DFHSTUP name	Description
Put/Putq main storage requests	The number of records that application programs wrote to main temporary storage.
Get/Getq main storage requests	The number of records that application programs obtained from main temporary storage.
Current TSMAINLIMIT setting	The current limit for the amount of storage that CICS makes available for data in main temporary storage.
Times at TSMAINLIMIT	The number of times that main temporary storage use attempted to exceed the limit for the amount of storage allowed for data.
Peak storage used for TSMAINLIMIT	The peak amount of storage that was used for data in main temporary storage.
Number of queues auto deleted	The number of temporary storage queues that CICS has deleted automatically by using the clean up task.
Count of clean up task runs	The number of times that the clean up task, which deletes eligible temporary storage queues automatically, has run.
Put/Putq auxiliary storage requests	The number of records that application programs wrote to auxiliary temporary storage.
Get/Getq auxiliary storage requests	The number of records that application programs obtained from auxiliary temporary storage.
Peak temporary storage names in use	The peak number of temporary storage queue names in use at any one time.

Table 191. Temporary storage: Summary global statistics (continued)

DFHSTUP name	Description
Number of entries in longest queue	The peak number of items in any one temporary storage queue, up to a maximum of 32767.
Times queues created	The number of times that CICS created individual temporary storage queues.
Control interval size	The size of the VSAM 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 control intervals (CIs) permits more data to be transferred at one time, resulting in less system overhead.
Available bytes per control interval	The number of bytes available for use in the temporary storage data set control interval.
Segments per control interval	The number of segments available in each temporary storage data set control interval.
Bytes per segment	The number of bytes per segment of the temporary storage data set.
Writes more than control interval	The 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	The size, expressed in bytes, of the longest record written to the temporary storage data set.
Number of control intervals available	The number of control intervals (CIs) available for auxiliary temporary 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 in use	The peak number of control intervals (CIs) that contain active data.
Times aux. storage exhausted	The number of situations where one or more transactions might have been suspended because of a NOSPACE condition, or might have been forced to end abnormally (by using a HANDLE CONDITION NOSPACE command). If statistics are present for this field, increase the size of the temporary storage data set.
Number of temp. storage compressions	The number of times that the temporary storage buffers were compressed.
Temporary storage buffers	The number of temporary storage buffers specified in the TS= system initialization parameter, or in the overrides. The number of buffers allocated might exceed the number requested.
Buffer waits	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.

Table 191. Temporary storage: Summary global statistics (continued)

DFHSTUP name	Description
Peak users waiting on buffers	The peak number of requests queued because no buffers were available.
Buffer writes	The number of WRITEs to the temporary storage data set. This includes both WRITEs required for recovery (see Forced writes for recovery) and WRITEs required when the buffer is needed to accommodate another control interval (CI). To minimize input/output activity caused by the second situation, increase buffer allocation.
Forced writes for recovery	The subset of the total number of WRITEs caused by recovery being specified for queues. This input/output activity is not affected by buffer allocation.
Buffer reads	The number of times a control interval (CI) must be read from disk. To decrease this activity, increase the buffer allocation.
Format writes	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 data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used.
Temporary storage strings	The number of temporary storage strings specified in the TS= system initialization parameter, or in the overrides. The number of strings allocated might exceed the number requested.
Peak number of strings in use	The peak number of concurrent input/output operations. If this is significantly less than the number specified in the system initialization table (SIT), consider reducing the SIT value to approach this number.
Times string wait occurred	The number of input/output requests that were queued because no strings were available. If the number of strings is the same as the number of buffers, this number is zero. If this number is a high percentage (over 30%) of the total number of input/output requests (for this purpose, the sum of TSGTWTN, Buffer writes, and TSGTRDN, Buffer reads), consider increasing the number of strings initially allocated.
Peak number of users waiting on string	The peak number of input/output requests that were queued at any one time because all strings were in use.
I/O errors on TS data set	The number of input/output errors that occurred on the temporary storage data set. Normally, this number should be zero. If it is not, inspect the CICS and VSAM messages to determine the cause.
Shared pools defined	The number of unique shared TS queue pools defined to CICS.
Shared pools currently connected	The number of the shared TS pools that this CICS region is connected to.
Shared read requests	The number of TS READQs from the Shared TS Queue pool of TS queues.
Shared write requests	The number of TS WRITEQs to the Shared TS Queue pool of TS queues.

Terminal control 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 users. Line-transmission faults and transaction faults are shown (these both have a negative influence on performance behavior).

Terminal control: Resource statistics

You can retrieve terminal control: resource statistics by using the **EXEC CICS COLLECT STATISTICS TERMINAL** system command. They are mapped by the DFHA06DS DSECT.

These statistics are gathered for each terminal, including ISC and IRC (MRO) sessions.

In addition to this, this DSECT should be used to map the terminal totals record.

Table 192. Terminal control: Resource statistics

DFHSTUP name	Field name	Description
Term Id	A06TETI	is the identifier of each terminal, which may have been statically defined, autoinstalled, or generated from the SESSIONS definition for a connection. Reset characteristic: not reset
LUname	A06LUNAM	is the terminal LU name. Reset characteristic: not reset
Terminal Type	A06TETT	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the *CICS Application Programming Reference*. . Reset characteristic: not reset
Acc Meth	A06EAMIB	is the terminal access method as defined in the TCT. This may be “SNA1”, “MRO”, “GAM”, “SNA2”, “BSAM”, or “VTAM” (now the z/OS Communications Server). For more information about access methods and their codes, see the DFHTCTTE DSECT in the *CICS Data Areas* guide. Reset characteristic: not reset
Conn ID	A06SYSID	is the owning connection name of this terminal/session. Reset characteristic: not reset
No. of Xactions	A06TEOT	is the number of transactions, both conversational 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.

Table 192. Terminal control: Resource statistics (continued)

DFHSTUP name	Field name	Description
Xaction 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.
Storage Viols	A06CSVC	is the number of storage violations that have occurred on this terminal. Reset characteristic: reset to zero
Input Messages For more information see1 on page 724	A06TENI	See note. Reset characteristic: reset to zero
Output Messages For more information see 1 on page 724	A06TENO	See note. Reset characteristic: reset to zero
Xmission Errors	A06TETE	is the number of errors for this terminal, or the number of disconnects for this session. Reset characteristic: reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06TCNT	is the total throwaway count. Reset characteristic: reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06SCNT	is the number of consecutive throwaways. Reset characteristic: reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06MCNT	is the maximum throwaway count. Reset characteristic: reset to zero
Pipeline Message: NOT IN THE DFHSTUP REPORT	A06PRTY	is the terminal priority Reset characteristic: not reset
Pipeline Message: TIOA Storage	A06STG	is the TIOA storage allowed at this terminal. Reset characteristic: reset to zero
Autoinstall Time: Logon	A06ONTM	is time at which this terminal/session was autoinstalled. This time is expressed as *hours:minutes:seconds.decimals*. The DSECT field contains the value as a store clock (STCK) value in local time. Reset characteristic: not reset
Autoinstall Time: Logoff	A06OFFTM	is the time at which this terminal/session 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 local time. Note that this field is only set on an Unsolicited Statistics (USS) record. Reset characteristic: not reset

Table 192. Terminal control: Resource statistics (continued)

DFHSTUP name	Field name	Description
Autoinstall Time: NOT IN THE DFHSTUP REPORT	A06GONTM	is the time at which this terminal/session was autoinstalled. The DSECT field contains the value as a store clock (STCK) value in GMT. Reset characteristic: not reset
Autoinstall Time: NOT IN THE DFHSTUP REPORT	A06GOFTM	is the time at which this terminal/session was logged off. The DSECT field contains the value as a store clock (STCK) value in GMT. Note that this field is only set on an Unsolicited Statistics (USS) record. Reset characteristic: not reset

Note:

1. 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.

Related reference:

COLLECT STATISTICS in Reference > System programming

Terminal control: Summary resource statistics

Terminal control summary resource statistics are not available online.

Table 193. Terminal control: Summary resource statistics

DFHSTUP name	Description
Term Id	is the identifier of each terminal, which may have been statically defined, autoinstalled, or generated from the SESSIONS definition for a connection.
LUname	is the terminal LU name.
Terminal Type	is the terminal type as defined in the TCT. For information about terminal types and their codes, see the*CICS Application Programming Reference*.
Acc Meth	is the terminal access method as defined in the TCT. This may be “SNA1”, “MRO”, “GAM”, “SNA2”, “BSAM”, or “VTAM” (now z/OS Communications Server). For more information about access methods and their codes, see the DFHTCTTE DSECT in the *CICS Data Areas* guide.
Conn ID	is the last value found for the owning connection name for this terminal/session.
No. of Xactions	is the number of transactions, both conversational 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.

Table 193. Terminal control: Summary resource statistics (continued)

DFHSTUP name	Description
Xaction Errors	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. 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.
Storage Viols	is the number of storage violations that have occurred on this terminal.
Input Messages	See note.
Output Messages	See note.
Xmission Errors	is the number of errors for this terminal, or the number of disconnects for this session.
Pipeline Message: Avg TIOA Storage	is the average TIOA storage used by this terminal.
Pipeline Message: Avg logged on time	is the average logged on time for an autoinstalled terminal/session. This field is blank if the terminal/session is not autoinstalled.

Note: Input messages 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.

Transaction class (TCLASS) statistics

Related concepts:

"Interpreting transaction class (TRANCLASS) statistics" on page 743
If you are never at the limit of your transaction class setting, you might consider resetting its value, or review whether there is any need to continue specifying any transaction types with that class.

Related reference:

"Transaction Classes report" on page 912
The Transaction Classes report is produced using a combination of the **EXEC CICS INQUIRE TRANCLASS** and **EXEC CICS EXTRACT STATISTICS TRANCLASS** commands.

Transaction class: resource statistics

You can retrieve transaction class resource statistics by using the **EXEC CICS EXTRACT STATISTICS TRANCLASS** system command. They are mapped by the DFHXMCDS DSECT.

Table 194. Transaction class: resource statistics

DFHSTUP name	Field name	Description
Tclass Name	XMCTCL	The 8-character name of the transaction class. Reset characteristic: not reset
Number Trandfs	XMCITD	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 transaction classes. Reset characteristic: not reset
Max Act	XMCMXT	The maximum number of transactions in the named transaction class that may be active concurrently. Reset characteristic: not reset
Purge Thresh	XMCTH	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	The total number of attach requests made for transactions in this transaction class. Reset characteristic: reset to zero
–AcptImm	XMCAI	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	The number of transactions that were purged immediately because the queue reached the purge threshold for this transaction class. Reset characteristic: reset to zero

Table 194. Transaction class: resource statistics (continued)

DFHSTUP name	Field name	Description
–Queued	XMCTQ	The total number of transaction that have queued for this transaction class. Reset characteristic: reset to zero
NOT IN THE DFHSTUP REPORT	XMCAAQ	The number of transactions that have become active in this transaction class but queued first. Reset characteristic: reset to zero
–PrgQ'd	XMCPWQ	The number of transactions that have been purged while 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	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	The highest number of active transactions reached in the transaction class. Reset characteristic: reset to current value
Peak Queued	XMCPQT	The highest number of transactions queued waiting for admittance to the transaction class. Reset characteristic: reset to current value
Times MaxAct	XMCTAMA	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 transaction class is zero and there are no active transactions in the transaction class. Reset characteristic: reset to zero or one if transaction class is currently at its maxactive limit.

Table 194. Transaction class: resource statistics (continued)

DFHSTUP name	Field name	Description
Times PrgThr	XMCTAPT	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	The current number of transactions currently active in this transaction class. Reset characteristic: not reset
–Queued	XMCCQT	The number of transactions that are currently queuing in this transaction class. Reset characteristic: not reset
–Queue Time	XMCCQTME	The total time in STCK units spent waiting by those transactions that are currently queuing in this transaction class. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset

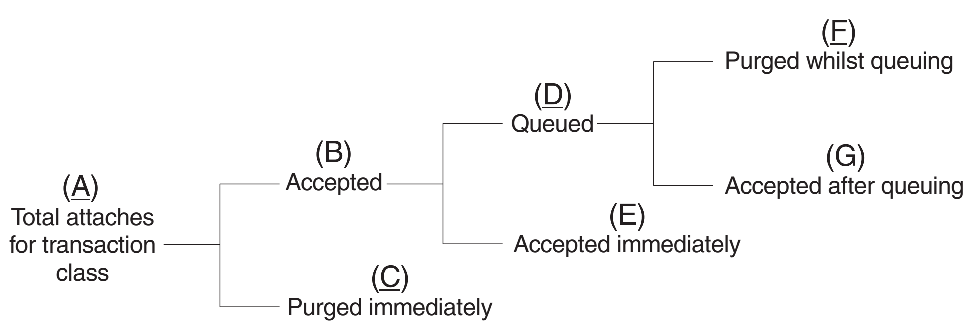
Table 194. Transaction class: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	XMC_TCLASS_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	XMC_TCLASS_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Figure 60 illustrates the transaction class statistics.



Attaches for Transaction class	=A		(XMCTAT)
Accepted	=B	(A - C)	
Purged immediately	=C		(XMCPI)
Queued	=D	(B - E)	
Accepted immediately	=E	(B - D)	(XMCAI)
Purged whilst queuing	=F		(XMCPWQ)
Accepted after queuing	=G	(D - F)	(XMCAAQ)

Figure 60. The transaction class statistics

Related information:

EXTRACT STATISTICS in Reference > System programming

Transaction class: Summary resource statistics

Transaction class summary resource statistics are not available online.

Table 195. Transaction class: Summary resource statistics

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 while 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 highest number of active transactions reached in the transaction class.
Peak Queued	The highest number of transactions queued waiting for admittance to the transaction class.
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 statistics

Related concepts:

"Interpreting transaction statistics" on page 734
Use these statistics to find out which transactions (if any) had storage violations.

"Interpreting transaction manager statistics"
The "Times the MAXTASK limit reached" indicates whether MXT is constraining your system, or any possible integrity exposures are resulting from forced resolutions of UOWs relating to the transactions. The only time that you must constrain your system in this way is to reduce virtual storage usage.

Related reference:

"Transaction Manager report" on page 913
The Transaction Manager report is produced using the **EXEC CICS EXTRACT STATISTICS TRANSACTION** command.

"Transactions report" on page 910
The Transactions report is produced using a combination of the **EXEC CICS INQUIRE TRANSACTION** and **EXEC CICS EXTRACT STATISTICS TRANSACTION** commands. The statistics data is mapped by the **DFHXMRDS**.

"Transaction Totals report" on page 915
The Transactions Totals report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command.

Interpreting transaction manager statistics

The "Times the MAXTASK limit reached" indicates whether MXT is constraining your system, or any possible integrity exposures are resulting from forced resolutions of UOWs relating to the transactions. The only time that you must constrain your system in this way is to reduce virtual storage usage.

As most CICS virtual storage is above the 16 MB line you may be able to run your system without MXT constraints, but note that CICS does preallocate storage, above and below the 16 MB line, for each MXT whether it is used. Changing MXT affects your calculations for the dynamic storage areas. See "Setting the maximum task specification (MXT)" on page 71 for more information.

Transaction manager: Global statistics

You can retrieve transaction manager global statistics by using the **EXEC CICS EXTRACT STATISTICS TRANSACTION** system command. They are mapped by the DFHXMGDS DSECT.

Table 196. Transaction manager: Global statistics

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

Table 196. Transaction manager: Global statistics (continued)

DFHSTUP name	Field name	Description
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
Time MAXTASKS last changed	XMGLSMXT	is the date and time when the maximum number of user transactions (MXT) was last set or changed dynamically. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds:decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. 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
Time last transaction attached	XMGLTAT	is the date and time when the last user transaction was attached. The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds:decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been attached since the statistics were last reset. Reset characteristic: reset to zero.
Current number of MAXTASK queued user transactions	XMGCQT	is the current number of queued user transactions in the system. Note that this does not include transactions queueing for transaction class membership. Note that the current queueing time for these transactions is in field XMGCQTME. 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)

Table 196. Transaction manager: Global statistics (continued)

DFHSTUP name	Field name	Description
Time the MAXTASKS limit last reached	XMGLAMXT	is the date and time when the number of active user transactions last equalled the specified maximum number of user transactions (MXT). The DFHSTUP report expresses this time as *day/month/year hours:minutes:seconds:decimals*; however, the DSECT field contains the time as a store clock (STCK) value in local time. Reset characteristic: not reset.
Currently at MAXTASKS limit	XMGATMXT	Indicates whether the CICS region is currently at the maximum number of user transactions (MXT). Reset characteristic: not reset.
Peak number of MAXTASK queued user transactions	XMGPQT	is the peak number of MAXTASK queued user transactions reached in the system. Reset characteristic: reset to current value (XMGCQT)
Peak number of active user transactions	XMGPAT	is the number of user transactions that have become active. Reset characteristic: reset to current value (XMGCAT)
Total number of active user transactions	XMGTAT	is the total number of user transactions that have become active. Reset characteristic: reset to zero
Total number of MAXTASK delayed user transactions	XMGTDT	is the number of user transactions that had to queue for MXT reasons. This value does not include those transactions that are currently queueing for MXT (see XMGCQT). Note that the queueing time for these transactions is in field XMGTQTME. 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. This value does not include the time spent by those transactions that are currently queueing for MXT (see XMGCQTME). 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

Related information:

EXTRACT STATISTICS in Reference > System programming

Transactions: resource statistics

You can retrieve transaction resource statistics by using the **EXEC CICS EXTRACT STATISTICS TRANSACTION** system command. They are mapped by the DFHXMRDS DSECT.

There are two sections in the DFHSTUP report for transaction manager resource statistics:

- "Transactions: Resource statistics - resource information"
- "Transactions: Resource statistics - integrity information" on page 737

Related information:

EXTRACT STATISTICS in Reference > System programming

Interpreting 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.

Transactions: Resource statistics - resource information

The transaction statistics show how often each transaction is called.

Table 197. Transactions: resource statistics - resource information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	The transaction identifier associated with the transaction definition. Reset characteristic: not reset
Program Name	XMRPN	The name of the initial program to which the transaction linked. Reset characteristic: not reset
Tclass Name	XMRTCL	The name of the transaction class in which the transaction is defined. Reset characteristic: not reset
Prty	XMRPRTY	The priority of the transaction, from 0 - 255. Reset characteristic: not reset
Remote Name	XMRRNAM	The name of the transaction on the remote system. Reset characteristic: not reset

Table 197. Transactions: resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Remote Sysid	XMRRSYS	The name of the remote system where the transaction resides. Reset characteristic: not reset
Dynamic	XMRDYN	Indicates whether the transaction is defined as DYNAMIC=YES (Y) or DYNAMIC=NO (N). Reset characteristic: not reset
Attach Count	XMRAC	The number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction runs. Reset characteristic: reset to zero
Retry Count	XMRRC	The number of times that this transaction definition has been used to retry a transaction. Reset characteristic: reset to zero
Dynamic Local	XMRDLC	The number of times that the dynamic transaction routing exit chose 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 Writing a dynamic routing program in the *CICS Customization Guide*. Reset characteristic: reset to zero
Dynamic Remote	XMRDRC	The number of times that the dynamic transaction routing exit chose to run this transaction on a remote system. This field is zero if the transaction is not defined as DYNAMIC=YES. For further guidance about dynamic transaction routing, see the programming information in Writing a dynamic routing program in the *CICS Customization Guide*. Reset characteristic: reset to zero

Table 197. Transactions: resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Remote Starts	XMRRSC	The number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is counted only in the CICS region that initiates the process, and not in the remote system where the transaction runs. In some circumstances, the use of a transaction definition for a remote start is not counted. These circumstances include the case in which a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command. Reset characteristic: reset to zero
Storage Violations	XMRSVC	The number of storage violations for this transaction that have been detected by CICS storage management. This statistic raises a serious concern if it occurs in a production system. You must act immediately to identify the cause of the problem because it can lead to data corruption, and therefore cannot be allowed to continue in an operational system. Reset characteristic: reset to zero
Not in DFHSTUP report	XMR_TRAN_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset

Table 197. Transactions: resource statistics - resource information (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	XMR_TRAN_ENTRYPOINT	Whether the transaction is defined as an application entry point. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	XMR_TRAN_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Transactions: Resource statistics - integrity information

The integrity information statistics show the potential integrity exposures that may have occurred during transaction execution as a result of inabilities to shunt UOWs, or forcing of shunted UOWs to complete regardless of the decisions made by participating systems.

Table 198. Transactions: Resource statistics - integrity information

DFHSTUP name	Field name	Description
Trans ID	XMRTI	is the transaction identifier associated with the transaction definition. Reset characteristic: not reset

Table 198. Transactions: Resource statistics - integrity information (continued)

DFHSTUP name	Field name	Description
Indoubt Wait	XMRIWTOP	Is the indicator of whether the transaction has been defined to support Indoubt Waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronisation with its coordinator. The indoubt wait option can have the following settings: • XMRIWTY = 'Y' = Transaction can support waiting • XMRIWTN = 'N' = Transaction cannot support waiting. Reset characteristic: not reset
Indoubt Wait timeout	XMRITOV	Is the indoubt wait timeout limit defined for this transaction, specified in minutes. This value has meaning only if the transaction is also defined to be able to wait indoubt (see XMRIWTOP). A value of zero, specifies that there is no timeout should this transaction be shunted by the CICS recovery manager. Reset characteristic: not reset
Indoubt Action	XMRIACTN	Is an indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting that, the timeout value specified has expired. Both of these events will force a resolution of the UOW in the direction specified by this field. The values can be : • XMRIACOM = 'C' = UOW will syncpoint forwards • XMRIABCK = 'B' = UOW will syncpoint backwards (rollback) Reset characteristic: not reset
Indoubt Waits	XMRIWAIT	Is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction. Reset characteristic: not reset
Indoubt action forced: Trandefn	XMRFATXN	Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. XMRIWTOP = XMTIWTN). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: not reset

Table 198. Transactions: Resource statistics - integrity information (continued)

DFHSTUP name	Field name	Description
Indoubt action forced: Timeout	XMRFAIT	Is the number of times this transaction id had a UOW that, although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value (XMRITOV) had been exceeded. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: not reset
Indoubt action forced: Operator	XMRFAOP	Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or system command forced a resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Indoubt action forced: No waiting	XMRFANW	Is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, although the transaction definition specified that it could (XMRIWTOP = XMRIWTY), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero
Indoubt action forced: Other	XMRFAOT	Is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, for reasons other than those already referenced in this table. This could be, for example, a recovery coordinator which has been cold started, a resynchronization protocol violation or failure, or because the level of resource manager (RMI) adaptor has not yet been changed to support indoubt resolution. The UOW would have been forced to resolve in the direction specified by XMRIACTN, regardless of the actions taken by any other participating region in this distributed UOW. Reset characteristic: reset to zero

Table 198. Transactions: Resource statistics - integrity information (continued)

DFHSTUP name	Field name	Description
Action mismatch	XMRAMISM	is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the fields already described in this table), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies. Reset characteristic: reset to zero

Transaction manager: Summary global statistics

Transaction manager summary global statistics are not available online.

Table 199. Transaction manager: Summary global statistics

DFHSTUP name	Description
Total number of transactions (user + system)	is the total number of tasks that have run in the system.
MAXTASK limit	is the last MXT value (expressed as a number of tasks) that was specified in the SIT, or as an override, or changed dynamically using the **EXEC CICS SET SYSTEM MAXTASKS(fullword binary data-value)** command.
Time the MAXTASK limit last changed	is the date and time when the maximum number of user transactions (MXT) was last set or changed dynamically.
Times the MAXTASK limit reached	is the total number of times MXT has been reached.
Time the MAXTASK limit last reached	is the date and time when the number of active user transactions last equalled the specified maximum number of user transactions (MXT).
Peak number of MAXTASK queued user transactions	is the peak number of MAXTASK queued user transactions reached in the system.
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.

Table 199. Transaction manager: Summary global statistics (continued)

DFHSTUP name	Description
Average MAXTASK queuing time of queued transactions	is the average time spent waiting by those user transactions that had to queue for MXT reasons.

Transactions: Summary resource statistics - resource information

Transactions summary resource statistics - resource information are not available online.

Table 200. Transactions: Summary resource statistics - resource information

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.
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 number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction runs.
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 and programming information about dynamic transaction routing, see Writing a dynamic routing program in the *CICS 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 Writing a dynamic routing program in the *CICS Customization Guide*

Table 200. Transactions: Summary resource statistics - resource information (continued)

DFHSTUP name	Description
Remote Starts	is the number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is only counted in the CICS region that initiates the process, and not in the remote system where the transaction runs. In some circumstances, the use of a transaction definition for a remote start is not counted. This includes the case where a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value, is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command.
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.

Transactions: Summary resource statistics - integrity information

Transactions summary resource statistics - integrity information are not available online.

Table 201. Transactions: Summary resource statistics - integrity information

DFHSTUP name	Description
Trans ID	is the transaction identifier associated with the transaction definition.
Indoubt Wait	is the last value encountered for the indicator of whether the transaction has been defined to support indoubt waiting in the event of an two-phase commit indoubt window failure. This means the failing UOW will be shunted by the CICS recovery manager awaiting resynchronization with its coordinator.
Indoubt Wait timeout	is the last value encountered for the indoubt wait timeout limit defined for this transaction, specified in minutes. This value only has any meaning if the transaction is also defined to be able to wait indoubt (see 'Indoubt Wait'). A value of zero specifies that there is no timeout should this transaction be shunted by the CICS recovery manager.
Indoubt Action	is the last value encountered for the indicator of which way this transaction will commit its UOWs in the event of not being able to wait indoubt (shunted), when an indoubt wait failure occurs. Or if the transaction had been waiting, that the timeout value specified had expired. Both of these events will force a resolution of the UOW in the direction specified by this field.
Indoubt Waits	is the number of indoubt waits (shunts) that have occurred for UOWs executing on behalf of this transaction.

Table 201. Transactions: Summary resource statistics - integrity information (continued)

DFHSTUP name	Description
Indoubt action forced: Trandefn	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, because the transaction definition for this transaction id specified that it could not support indoubt waiting (ie. Indoubt Wait = No). The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Timeout	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because the indoubt wait timeout value had been exceeded. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Operator	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, because an operator (CEMT) or system command forced a resolution. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action' by default, or in the direction specified by the operator, regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: No waiting	is the number of times this transaction id had a UOW that could not be shunted when an indoubt failure occurred, even though the transaction definition specified that it could (Indoubt Wait = Yes), because the resource managers (RMIs) or CICS resources or CICS connections used by the UOW could not support indoubt waiting (shunting). The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Indoubt action forced: Other	is the number of times this transaction id had a UOW that although shunted because of an indoubt failure, had the wait for resynchronization with its recovery coordinator terminated prematurely, for reasons other than those already referenced in this table. This could be, for example, a cold started recovery coordinator, a resynchronization protocol violation or failure, or because the level of resource manager (RMI) adapter has not yet been changed to support indoubt resolution. The UOW would have been forced to resolve in the direction specified by 'Indoubt Action', regardless of the actions taken by any other participating region in this distributed UOW.
Action mismatch	is the number of times this transaction id had a UOW that was forced to resolve using the indoubt action attribute, whether by definition, option or operator override (as detailed in the fields already described in this table), and on doing so detected an indoubt action attribute mismatch with a participating system or resource manager (RMI). For example, a participating system in a distributed UOW resolves its work forward while other systems back out theirs. The opposite also applies.

Interpreting transaction class (TRANCLASS) statistics

If you are never at the limit of your transaction class setting, 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 “Transaction class (TCLASS) statistics” on page 725.

Transient data statistics

Related concepts:

“Interpreting transient data statistics”
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.

Related reference:

“Transient Data report” on page 916
The Transient Data report is produced using the **EXEC CICS EXTRACT STATISTICS TDQUEUE** command. The statistics data is mapped by the **DFHTQGDS**.

“Transient Data Queues report” on page 917
The Transient Data Queues report is produced using a combination of the **EXEC CICS INQUIRE TDQUEUE** and **EXEC CICS EXTRACT STATISTICS TDQUEUE** commands. The statistics data is mapped by the **DFHTQRDS DSECT**.

“Transient Data Queue Totals report” on page 919
The Transient Data Queues Totals report is produced using a combination of the **EXEC CICS INQUIRE TDQUEUE** and **EXEC CICS EXTRACT STATISTICS TDQUEUE** commands. The statistics data is mapped by the **DFHTQRDS DSECT**.

Interpreting transient data statistics

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.

Transient data: Global statistics

You can retrieve transient data global statistics by using the **EXEC CICS EXTRACT STATISTICS TDQUEUE** system command. They are mapped by the DFHTQGDS DSECT.

For more information on using transient data statistics, see Chapter 18, “CICS transient data (TD) facility: Performance and tuning,” on page 249.

Table 202. Transient data: Global statistics

DFHSTUP name	Field name	Description
Control interval size	TQGACISZ	is the size of the control interval, expressed in bytes. Reset characteristic: not reset

Table 202. Transient data: Global statistics (continued)

DFHSTUP name	Field name	Description
Control intervals	TQGANCIS	is the number of control intervals in the intrapartition data set DFHINTRA. Reset characteristic: not reset
Current control intervals in use	TQGACTCI	is the current number of control intervals in the intrapartition data set DFHINTRA. Reset characteristic: not reset
Peak control intervals used	TQGAMXCI	is the peak value of the number of control intervals concurrently active in the system. Reset characteristic: reset to current value
Times NOSPACE occurred	TQGANOSP	is the number of times that a NOSPACE condition has occurred. Reset characteristic: reset to zero
Writes to intrapartition data set	TQGACTPT	is the number of WRITEs to the intrapartition transient 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 intrapartition data set	TQGACTGT	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	TQGACTFT	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	TQGACTIO	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:

Table 202. Transient data: Global statistics (continued)

DFHSTUP name	Field name	Description
Intrapartition buffers	TQGANBFA	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
Current buffers containing valid data	TQGACNIU	is the current number of intrapartition buffers that contain valid data. Reset characteristic: not reset
Peak intra. buffers containing valid data	TQGAMXIU	is the peak number of intrapartition buffers which contain valid data. Reset characteristic: reset to current value
Intrapartition accesses	TQGATNAL	is the number of times intrapartition buffers have been accessed. Reset characteristic: reset to current value
Current concurrent buffer accesses	TQGACNAL	is the current value of the number of concurrent intrapartition buffer accesses. Reset characteristic: not reset
Peak concurrent intrapartition accesses	TQGAMXAL	is the peak value of the number of concurrent intrapartition buffer accesses. Reset characteristic: reset to current value
Intrapartition buffer waits	TQGATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also 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
Current intrapartition buffer waits	TQGACNWT	is the current number of requests queued because no buffers were available. Reset characteristic: not reset
Peak intrapartition buffer waits	TQGAMXWT	is the peak number of requests queued because no buffers were available. Reset characteristic: reset to current value

Table 202. Transient data: Global statistics (continued)

DFHSTUP name	Field name	Description
All of the intrapartition data set statistics referenced in the table are printed, even if the values reported are zero.		

CICS produces the following statistics for multiple strings:

DFHSTUP name	Field name	Description
Number of strings	TQGSNSTA	is the number of strings currently active. Reset characteristic: not reset
Times string accessed	TQGSTNAL	is the number of times a string was accessed. Reset characteristic: reset to current value
Current concurrent string accesses	TQGSCNAL	is the current number of strings concurrently accessed in the system. Reset characteristic: not reset
Peak concurrent string accesses	TQGSMXAL	is the peak number of strings concurrently accessed in the system. Reset characteristic: reset to current value
Intrapartition string waits	TQGSTNWT	is the number of times that tasks had to wait because no strings were available. Reset characteristic: reset to current value
Current intrapartition string waits	TQGSCNWT	is the current number of concurrent string waits in the system. Reset characteristic: not reset
Peak string waits	TQGSMXWT	is the peak number of concurrent string waits in the system. Reset characteristic: reset to current value

CICS produces the following statistics for buffer usage:

DFHSTUP name	Field name	Description
Intrapartition buffers	TQGANBFA	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

CICS produces the following statistics for buffer usage:

DFHSTUP name	Field name	Description
Current buffers containing valid data	TQGACNIU	is the current number of intrapartition buffers that contain valid data. Reset characteristic: not reset
Peak intra. buffers containing valid data	TQGAMXIU	is the peak number of intrapartition buffers which contain valid data. Reset characteristic: reset to current value
Intrapartition accesses	TQGATNAL	is the number of times intrapartition buffers have been accessed. Reset characteristic: reset to current value
Current concurrent buffer accesses	TQGACNAL	is the current value of the number of concurrent intrapartition buffer accesses. Reset characteristic: not reset
Peak concurrent intrapartition accesses	TQGAMXAL	is the peak value of the number of concurrent intrapartition buffer accesses. Reset characteristic: reset to current value
Intrapartition buffer waits	TQGATNWT	is the number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also 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
Current intrapartition buffer waits	TQGACNWT	is the current number of requests queued because no buffers were available. Reset characteristic: not reset
Peak intrapartition buffer waits	TQGAMXWT	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 referenced in the table are printed, even if the values reported are zero.

CICS produces the following statistics for multiple strings:

DFHSTUP name	Field name	Description
Number of strings	TQGSNSTA	is the number of strings currently active. Reset characteristic: not reset
Times string accessed	TQGSTNAL	is the number of times a string was accessed. Reset characteristic: reset to current value
Current concurrent string accesses	TQGSCNAL	is the current number of strings concurrently accessed in the system. Reset characteristic: not reset
Peak concurrent string accesses	TQGSMXAL	is the peak number of strings concurrently accessed in the system. Reset characteristic: reset to current value
Intrapartition string waits	TQGSTNWT	is the number of times that tasks had to wait because no strings were available. Reset characteristic: reset to current value
Current intrapartition string waits	TQGSCNWT	is the current number of concurrent string waits in the system. Reset characteristic: not reset
Peak string waits	TQGSMXWT	is the peak number of concurrent string waits in the system. Reset characteristic: reset to current value

Related information:

EXTRACT STATISTICS in Reference > System programming

Transient data: resource statistics

You can retrieve transient data resource statistics by using the **EXEC CICS EXTRACT STATISTICS TDQUEUE** system command. They are mapped by the DFHTQRDS DSECT.

Transient data resource statistics are collected for each queue. You can use the information from the statistics for each queue to calculate the average number of transient data accesses per transaction. The items in this listing reflect the information you placed in the definition for the transient data queue.

The TQRQTYPE field is not displayed in the DFHSTUP report. It signifies the queue type, which can be one of the following fields:

- TQRQTEXT (X'01') for extrapartition queues

- TQRQTINT (X'02') for intrapartition queues
- TQRQTIND (X'03') for indirect queues
- TQRQTREM (X'04') for remote queues.

TQRQTYPE is reset to zero.

Related information:

EXTRACT STATISTICS in Reference > System programming

Transient data: Resource statistics - intrapartition transient data queues

Table 203. Transient data: Resource statistics - intrapartition transient data queues

DFHSTUP name	Field name	Description
Queue id	TQRQID	The destination identifier (queue) that you specified in the transient data queue definition. Reset characteristic: Not reset
Request Counts: Number of Writes	TQRWRITE	The total number of requests to write to this queue. Reset characteristic: Reset to zero
Request Counts: Number of Reads	TQRREAD	The total number of requests to read from this queue. Reset characteristic: Reset to zero
Request Counts: Number of Deletes	TQRDELET	The total number of requests to delete this queue. Reset characteristic: Reset to zero
ATI Information: Trigger level	TQRTRIGL	The value of the ATI trigger level. If the number of items in this queue reaches this value the transaction id in TQRATRAN is attached to process the items in the queue. Reset characteristic: Not reset
ATI Information: Tran Id	TQRATRAN	The id of the transaction that will be scheduled against a terminal or session or in the background (see TQRFTYPE) when the trigger level (TQRTRIGL) has been reached. Reset characteristic: Not reset

Table 203. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
ATI Information: Facility Type	TQRFTYPE	The ATI facility type for this transient data queue. This will be where and how the transaction id in TQRATRAN is attached when the ATI trigger level (TQRTRIGL) is reached. It can have the following values: • TQRFTNA X'00' Not Applicable (N/A) • TQRFTTRM X'01' Terminal (TERM) • TQRFTSYS X'02' System (SYS) • TQRFTNTE X'03' No terminal (NONE). Reset characteristic: Not reset
ATI Information: Facility Name	TQRFNAME	The id of the system or terminal that the trigger transaction will be attached against. This value is blank when there is no facility. Reset characteristic: Not reset
ATI Information: No. of triggers	TQRTRIGN	The number of times the trigger transaction (TQRATRAN) has been scheduled, as a result of the trigger level (TQRTRIGL) being exceeded. Reset characteristic: Reset to zero
Recovery: Rcvy type	TQRRTYPE	The recoverable type of this transient data queue. It can have the following values: • TQRRTNA X'00' Not applicable (N/A) • TQRRTPH X'01' Physical recoverable (PH) • TQRRTLG X'02' Logical recoverable (LG) • TQRRTNR X'03' Non-recoverable (NR) Reset characteristic: Not reset

Table 203. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
Recovery: Wait opt.	TQRWAIT	Indicates whether any transactions that use this queue can, if they lose the connection to their recovery coordinator, wait indoubt (shunted). If the queue supports indoubt waiting (TQRWTYES), the locks that are associated with that UOW will be held until syncpoint resolution. If not, the UOW will be committed (forward or backward) at the time of indoubt failure, according to the settings in the transaction definition, and the locks released as a result. This field has meaning only if the queue is logically recoverable. The indoubt wait option can have the following settings: • TQRWTNA X'00' Not Applicable (N/A) • TQRWTYES X'01' Queue supports indoubt waiting (YES) • TQRWTNO X'02' Does not support indoubt waiting (NO) Reset characteristic: Not reset
Recovery: Wait Action	TQRWAITA	Indicates whether this transient data queue will reject or suspend subsequent requests to this queue. This can be when a UOW that has used this queue has been shunted because of an indoubt failure and is therefore retaining enqueues against this queue. This field has no meaning if the queue is non-recoverable or does not support indoubt waiting (see TQRWAIT). The possible values for this field are: • TQRWANA X'00' Not Applicable (N/A) • TQRWAREJ X'01' Further requests will be rejected (REJECT) • TQRWAQUE X'02' Further requests will be queued (QUEUE) Reset characteristic: Not reset
DFHINTRA usage: Current CIs used	TQRCCIUS	The number of control intervals (CIs) that are currently in use on the DFHINTRA data set by this queue. Reset characteristic: Not reset
DFHINTRA usage: Peak CIs used	TQRPCIUS	The peak number of control intervals (CIs) that have been used on the DFHINTRA data set by this queue. Reset characteristic: Reset to current

Table 203. Transient data: Resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
DFHINTRA usage: Current items	TQRCNITM	The current number of items in this intrapartition queue. Reset characteristic: Not reset
DFHINTRA usage: Peak items	TQRPNITM	The peak number of items in this intrapartition queue. Reset characteristic: Reset to current
Not in DFHSTUP report	TQR_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID.

For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Transient data: Resource statistics - extrapartition transient data queues

Table 204. Transient data: Resource statistics - extrapartition transient data queues

DFHSTUP name	Field name	Description
Queue ID	TQRQID	The destination identifier (queue) that you specified in the transient data queue definition. Reset characteristic: Not reset
DD name (assoc.)	TQRDDNM	The associated DD name of this data set in the CICS start-up JCL. Reset characteristic: Not reset
Data set name (Destination/origin of data)	TQRDSNNM	The data set name of the extrapartition transient data queue. Reset characteristic: Not reset
Member Name	TQRPDSMN	The name of a member in the partitioned data set referenced by the ddname for the extrapartition transient data queue. Reset characteristic: Not reset
I/O Type	TQRIOTYP	Is an indicator of the input/output type of the extrapartition data set. It might contain one of the following values: • TQRIONA X'00' Not Applicable • TQRIOIN X'01' Input • TQRIOOUT X'02' Output • TQRIORDB X'03' Readback (input but read backwards) Reset characteristic: Not reset
No. of Writes	TQRWRITE	The total number of write operations to the output data set. Reset characteristic: Reset to zero
No. of Reads	TQRREAD	The total number of read operations from the input data set. Reset characteristic: Reset to zero

Table 204. Transient data: Resource statistics - extrapartition transient data queues (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	TQR_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Transient data: Resource statistics - indirect transient data queues

Table 205. Transient data: Resource statistics - indirect transient data queues

DFHSTUP name	Field name	Description
Queue ID	TQRQID	The destination identifier (queue) that you specified in the transient data queue definition. Reset characteristic: Not reset
Indirect Queue id	TQRIQID	The name of the indirect queue. Reset characteristic: Not reset
Request Counts: Writes	TQRWRITE	The total number of requests to write to this queue. Reset characteristic: Reset to zero
Request Counts: Reads	TQRREAD	The total number of requests to read from this queue. Reset characteristic: Reset to zero
Request Counts: Deletes	TQRDELET	The total number of requests to delete this queue.. Reset characteristic: Reset to zero
Not in DFHSTUP report	TQR_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset

Table 205. Transient data: Resource statistics - indirect transient data queues (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	TQR_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Transient data: Resource statistics - remote transient data queues

Table 206. Transient data: Resource statistics - remote transient data queues

DFHSTUP name	Field name	Description
Queue Id	TQRQID	The destination identifier (queue) that you specified in the transient data queue definition. Reset characteristic: Not reset
Remote: Queue	TQRRQID	The name of the queue on the remote system (TQRRSYS). Reset characteristic: Not reset
Remote: Sysid	TQRRSYS	The connection id of the CICS system that owns this queue. Reset characteristic: Not reset
Request Counts: Writes	TQRWRITE	The total number of requests to write to this queue. Reset characteristic: Reset to zero

Table 206. Transient data: Resource statistics - remote transient data queues (continued)

DFHSTUP name	Field name	Description
Request Counts: Reads	TQRREAD	The total number of requests to read from this queue. Reset characteristic: Reset to zero
Request Counts: Deletes	TQRDELET	The total number of requests to delete this queue. Reset characteristic: Reset to zero
Not in DFHSTUP report	TQR_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	TQR_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	TQR_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource

signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Transient data: Summary global statistics

Transient data summary global statistics are not available online.

Table 207. Transient data: Summary global statistics. **In the statistics produced for the intrapartition data set:**

DFHSTUP name	Description
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 intrapartition data set	is the total number of WRITEs to the transient data data set. This includes both WRITEs needed for recovery 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 intrapartition data set	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:

DFHSTUP name	Description
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 buffers which contain valid data.
Intrapartition accesses	is the total number of times that intrapartition buffers have been accessed.
Peak concurrent intrapartition accesses	is the peak number of concurrent intrapartition buffer 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 also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available.

In the statistics produced for buffer usage:

DFHSTUP name	Description
Peak intrapartition buffer waits	is the peak number of requests queued because no buffers were available.

All of the intrapartition data set statistics referenced in the table are printed, even if the values reported are zero.

CICS produces the following statistics for multiple strings:

DFHSTUP name	Description
Times string 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.
Intrapartition string waits	is the total number of times that tasks had to wait because no strings were available.
Peak string waits	is the peak number of concurrent string waits in the system.

Transient data: Summary resource statistics

Transient data: Summary resource statistics are not available online.

Table 208. Transient data: Summary resource statistics - intrapartition transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.
Request Counts: Number of Writes	is the total number of requests to write to this queue.
Request Counts: Number of Reads	is the total number of requests to read from this queue.
Request Counts: Number of Deletes	is the total number of requests to delete this queue.
ATI Information: Trigger level	is the value of the ATI trigger level. If the number of items in this queue reaches this value, the transaction id in 'Tran Id' is attached to process the items in the queue.
ATI Information: Tran Id	is the id of the transaction that will be scheduled against a terminal/session or in the background (depending on the value of 'Facility Type'), when the trigger level ('Trigger level') has been reached.
ATI Information: Facility Type	is the ATI facility type for this transient data queue. This will be where and how the transaction id in 'Tran Id' is attached when the ATI trigger level ('Trigger level') is reached. It can have the following values:- • N/A — Not Applicable • TERM — Terminal • SYS — System • NONE — No terminal.
ATI Information: Facility Name	is the id of the system or terminal that the trigger transaction will be attached against. This value is blank when there is no facility.

Table 208. Transient data: Summary resource statistics - intrapartition transient data queues (continued)

DFHSTUP name	Description
ATI Information: No. of triggers	is the number of times the trigger transaction ('Tran Id') has been scheduled, as a result of the trigger level ('Trigger level') being exceeded.
Recovery: Rcvy type	is the recoverable type of this transient data queue. It can have the following values:- • N/A — Not applicable • PH — Physical recoverable • LG — Logical recoverable • NR — Non-recoverable
Recovery: Wait opt.	is an indicator of whether any transactions that use this queue will be able, in the event of losing the connection to their recovery coordinator, to wait indoubt (shunted). If the queue supports indoubt waiting (Wait opt. = Yes) then the locks that are associated with that UOW will be held until syncpoint resolution. If not, the UOW will be committed (forward or backward) at the time of indoubt failure according to the settings in the transaction definition and the locks released as a result. This field has meaning only if the queue is logically recoverable. The indoubt wait option can have the following settings: • N/A — Not Applicable • Yes — Queue supports indoubt waiting • No — Does not support indoubt waiting
Recovery: Wait Action	is an indicator of whether this transient data queue will reject or suspend subsequent requests to this queue. This can be when a UOW that has used this queue has been shunted because of an indoubt failure and is therefore retaining enqueues against this queue. This field has no meaning if the queue is non-recoverable (Rcvy Type is NR), or does not support indoubt waiting (Wait opt. is No). The possible values for this field are: • N/A — Not Applicable • Reject — Further requests will be rejected • Queue — Further requests will be queued
DFHINTRA usage: Current CIs used	is the current number of CIs used by this intrapartition queue.
DFHINTRA usage: Peak CIs used	is the peak number of CIs used by this intrapartition queue.
DFHINTRA usage: Current items	is the current number of items in this intrapartition queue.
DFHINTRA usage: Peak items	is the peak number of items in this intrapartition queue.

Table 209. Transient data: Summary resource statistics - extrapartition transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.
DDNAME (assoc.)	is the DDNAME of the extrapartition queue.

Table 209. Transient data: Summary resource statistics - extrapartition transient data queues (continued)

DFHSTUP name	Description
Data set name (Destination/origin of data)	is the data set name of the extrapartition queue.
Member Name	is the name of a member in the partitioned data referenced by the ddname for the extrapartition transient data queue.
I/O Type	is the type of I/O data set. Can be one of input, output or readback.
No. of Writes	is the total number of write operations to the output data set.
No. of Reads	is the total number of read operations from the input data set.

Table 210. Transient data: Summary resource statistics - indirect transient data queues

DFHSTUP name	Description
Queue ID	is the destination identifier (queue) that you specified in the transient data queue definition.
Indirect Queue id	is the name of the indirect queue.
Request Counts: Writes	is the total number of requests to write to this queue.
Request Counts: Reads	is the total number of requests to read from this queue.
Request Counts: Deletes	is the total number of requests to delete this queue.

Table 211. Transient data: Summary resource statistics - remote transient data queues

DFHSTUP name	Description
Queue Id	is the destination identifier (queue) that you specified in the transient data queue definition.
Remote: Queue	is the name of the remote queue.
Remote: Sysid	is the name of the remote system.
Request Counts: Writes	is the total number of requests to write to this queue.
Request Counts: Reads	is the total number of requests to read from this queue.
Request Counts: Deletes	is the total number of requests to delete this queue.

URIMAP definition statistics

URIMAP resource definitions match the URIs of HTTP or web service requests, and provide information about how to process the requests. The statistics include global statistics and statistics for each URIMAP definition.

DFH0STAT reports: See URIMAP global report and URIMAP report

Related reference:

"URIMAPs Global report" on page 919
The URIMAPs Global report is produced using the **EXEC CICS EXTRACT STATISTICS URIMAP** command. The statistics data is mapped by the **DFHWBGDS DSECT**.

"URIMAPs report" on page 920
The URIMAPs report is produced using a combination of **EXEC CICS INQUIRE URIMAP** and **EXEC CICS EXTRACT STATISTICS URIMAP RESID()** commands. The statistics data is mapped by the DFHWBRDS DSECT.

"Virtual Hosts report" on page 926
The Virtual Hosts report is produced using the **EXEC CICS INQUIRE HOST** command.

URIMAP definitions: Global statistics

You can retrieve URIMAP definition global statistics by using the **EXEC CICS EXTRACT STATISTICS URIMAP** system command. They are mapped by the DFHWBGDS DSECT.

Table 212. URIMAP definitions: Global statistics

DFHSTUP name	Field name	Description
URIMAP reference count	WBG_URIMAP_REFERENCE_COUNT	Number of times a search for a matching URIMAP definition was made. Reset characteristic: reset to zero
Entry point reference count	WBG_URIMAP_ENTRYPOINT_REF	Number of times a search for a matching URIMAP definition that is defined as an application entry point was made. Reset characteristic: reset to zero
Disabled	WBG_URIMAP_MATCH_DISABLED	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled. Reset characteristic: reset to zero
Host/Path no match count	WBG_URIMAP_NO_MATCH_COUNT	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found. Reset characteristic: reset to zero
Host/Path match count	WBG_URIMAP_MATCH_COUNT	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found. Reset characteristic: reset to zero

Table 212. URIMAP definitions: Global statistics (continued)

DFHSTUP name	Field name	Description
Redirected	WBG_URIMAP_MATCH_REDIRECT	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected. Reset characteristic: reset to zero
Analyzer used	WBG_URIMAP_MATCH_ANALYZER	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called. Reset characteristic: reset to zero
Static content delivered	WBG_URIMAP_STATIC_CONTENT	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or zFS file) was delivered as a response. Reset characteristic: reset to zero
Dynamic content delivered	WBG_URIMAP_DYNAMIC_CONTENT	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response. Reset characteristic: reset to zero
PIPELINE requests	WBG_URIMAP_PIPELINE_REQS	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a web service. Reset characteristic: reset to zero
ATOMSERVICE requests	WBG_URIMAP_ATOMSERV_REQS	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a Atom service. Reset characteristic: reset to zero
Scheme (HTTP) requests	WBG_URIMAP_SCHEME_HTTP	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP. Reset characteristic: reset to zero

Table 212. URIMAP definitions: Global statistics (continued)

DFHSTUP name	Field name	Description
Scheme (HTTPS) requests	WBG_URIMAP_SCHEME_HTTPS	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL). Reset characteristic: reset to zero
Virtual host disabled count	WBG_HOST_DISABLED_COUNT	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled. Reset characteristic: reset to zero
Direct attach count	WBG_URIMAP_DIRECT_ATTACH	Number of requests that are processed by directly attached user task. Reset characteristic: reset to zero

Related information:

EXTRACT STATISTICS in Reference > System programming

URIMAP definitions: Resource statistics

You can retrieve URIMAP definition resource statistics by using the **EXEC CICS EXTRACT STATISTICS URIMAP** system command. They are mapped by the DFHWBRDS DSECT.

The resource information gives details of various attribute settings of each URIMAP resource.

Table 213. URIMAP definitions: resource statistics

DFHSTUP name	Field name	Description
URIMAP Name	WBR_URIMAP_NAME	The name of the URIMAP definition. Reset characteristic: not reset

Table 213. URIMAP definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
URIMAP Usage	WBR_URIMAP_USAGE	The intended use of this URIMAP: **SERVER** The URIMAP definition is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH. **CLIENT** The URIMAP definition is used to specify information for making an HTTP request from CICS as an HTTP client. **PIPELINE** The URIMAP definition is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH. **ATOM** The URIMAP definition is used for an incoming request for data that CICS makes available as an Atom feed. **JVMSERVER** The URIMAP resource is used to map an inbound request from a web client to a servlet or JSP that is running in a JVM server. Reset characteristic: not reset
URIMAP Scheme	WBR_URIMAP_SCHEME	The scheme for the HTTP request, HTTP with SSL (HTTPS) or without (HTTP). Reset characteristic: not reset
Authenticate	WBR_URIMAP_AUTHENTICATE	For USAGE(CLIENT), whether credentials (authentication information) are sent for outbound web requests. Reset characteristic: not reset
URIMAP Port	WBR_URIMAP_PORT	For USAGE(CLIENT), the port number used for the client connection. For USAGE(SERVER), the port number that is being used for the communication, even if PORT(NO) is specified on the URIMAP at define time. Reset characteristic: not reset

Table 213. URIMAP definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
URIMAP Host	WBR_URIMAP_HOSTNAME	For the USAGE(CLIENT) option, the host name of the target URL to which the HTTP request is to be sent. For any other usage type, the host name on the incoming HTTP request that is used to select this URIMAP definition. Reset characteristic: not reset
URIMAP IP Family	WBR_URIMAP_IP_FAMILY	The address format of the IP Resolved Address. Reset characteristic: not reset
URIMAP IP Resolved Address	WBR_URIMAP_IP_ADDRESS	The IPv4 or IPv6 address of the host. Reset characteristic: not reset
URIMAP Path	WBR_URIMAP_PATH	For the USAGE(CLIENT) option, the path of the target URL to which the HTTP request is to be sent. For any other usage type, the path on the incoming HTTP request that is used to select this URIMAP definition. The path might end in an asterisk, meaning that it is generic, and matches any path with characters that are the same up to but excluding the asterisk. Reset characteristic: not reset
TCPIPSERVICE name	WBR_URIMAP_TCPIPSERVICE	The TCPIPSERVICE resource to which this URIMAP definition applies. Only requests received using this TCPIPSERVICE resource are matched to this URIMAP definition. If no TCPIPSERVICE resource is specified, the URIMAP definition applies to all incoming HTTP requests. Reset characteristic: not reset
WEBSERVICE name	WBR_URIMAP_WEBSERVICE	The name of the WEBSERVICE resource definition for the web service that handles the incoming HTTP request. Reset characteristic: not reset
PIPELINE name	WBR_URIMAP_PIPELINE	The name of the PIPELINE resource definition for the web service that handles the incoming HTTP request. Reset characteristic: not reset

Table 213. URIMAP definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
ATOMSERVICE name	WBR_URIMAP_ATOMSERVICE	The name of the ATOMSERVICE resource definition for the Atom document. Reset characteristic: not reset
Templatename	WBR_URIMAP_TEMPLATENAME	The name of a CICS document template, the contents of which are returned as the HTTP response. Reset characteristic: not reset
HFS file	WBR_URIMAP_HFSFILE	The name of a file in the z/OS UNIX System Services Hierarchical File System (HFS), with the contents that are returned as the HTTP response. Reset characteristic: not reset
Analyzer	WBR_URIMAP_ANALYZER_USE	Whether or not the analyzer associated with the TCPIPSERVICE definition is called to process the request. Reset characteristic: not reset
Converter	WBR_URIMAP_CONVERTER	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM. Reset characteristic: not reset
Transaction ID	WBR_URIMAP_TRANS_ID	The name of the alias transaction that processes the incoming HTTP request. Reset characteristic: not reset
Program name	WBR_URIMAP_PROGRAM_NAME	The name of the application program that processes the incoming HTTP request. Reset characteristic: not reset
Redirection type	WBR_URIMAP_REDIRECT_TYPE	Whether or not matching requests will be redirected, on a temporary or permanent basis. Reset characteristic: not reset

Table 213. URIMAP definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
Location for redirection	WBR_URIMAP_LOCATION	An alternative URL to which the Web client is redirected, if redirection is specified. Reset characteristic: not reset
URIMAP reference count	WBR_URIMAP_REFERENCE_COUNT	Number of times this URIMAP definition was referenced. Reset characteristic: reset to zero
Disabled	WBR_URIMAP_MATCH_DISABLED	Number of times this host and path were matched, but the URIMAP definition was disabled. Reset characteristic: reset to zero
Redirected	WBR_URIMAP_MATCH_REDIRECT	Number of times that this host and path were matched and the request was redirected. Reset characteristic: reset to zero
Time out for pooled sockets	WBR_URIMAP_SOCKETCLOSE	The time after which CICS discards pooled client HTTP connections created using this URIMAP resource if they are not reused. Reset characteristic: not reset
Number of pooled sockets	WBR_URIMAP_SOCKPOOLSIZE	Current number of open client HTTP connections held in the pool for reuse. Reset characteristic: not reset
Peak number of pooled sockets	WBR_URIMAP_SOCKPOOLSIZE_PEAK	Peak number of open client HTTP connections held in the pool for reuse. Reset characteristic: reset to zero
Number of reclaimed sockets	WBR_URIMAP_SOCKETS_RECLAIMED	Number of pooled connections that were closed in the pool by CICS because the CICS region had reached the MAXSOCKETS limit. Reset characteristic: reset to zero

Table 213. URIMAP definitions: resource statistics (continued)

DFHSTUP name	Field name	Description
Number of timed out sockets	WBR_URIMAP_SOCKETS_TIMEDOUT	Number of pooled connections that were closed in the pool by CICS because they reached their timeout value without being reused. Reset characteristic: reset to zero
Not in DFHSTUP report	WBR_URIMAP_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	WBR_URIMAP_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID,

DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

URIMAP definitions: summary global statistics

These global statistics show summary information and statistics about URIMAP resource definitions. Summary statistics are not available online.

Table 214. URIMAP definitions: summary global statistics

DFHSTUP name	Description
URIMAP reference count	Number of times a search for a matching URIMAP definition was made.
Entry point reference count	Number of times a search for a matching URIMAP definition that is defined as an application entry point was made.
Disabled	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled.
Redirected	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected.
Host/Path no match count	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found.
Host/Path match count	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found.
Analyzer used	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called.
Static content delivered	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or z/OS UNIX file) was delivered as a response.
Dynamic content delivered	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response.
PIPELINE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a web service.

Table 214. URIMAP definitions: summary global statistics (continued)

DFHSTUP name	Description
ATOMSERVICE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by an Atom service.
Scheme (HTTP) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP.
Scheme (HTTPS) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL).
Virtual host disabled count	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled.
Direct attach count	Number of requests that are processed by directly attached user task.

URIMAP definitions: Summary resource statistics

A summary listing of resource statistics for a URIMAP definition.

Summary statistics are not available online.

The resource information gives details of various attribute settings of each URIMAP definition.

Table 215. URIMAP definitions: summary resource statistics

DFHSTUP name	Description
URIMAP Name	The name of the installed URIMAP resource.

Table 215. URIMAP definitions: summary resource statistics (continued)

DFHSTUP name	Description
URIMAP Usage	The intended use of this URIMAP resource: **SERVER** The URIMAP resource is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH. **CLIENT** The URIMAP resource is used to specify information for making an HTTP request from CICS as an HTTP client. **PIPELINE** The URIMAP resource is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH. **ATOM** The URIMAP resource is used for an incoming request for data that CICS makes available as an Atom feed. **JVMSERVER** The URIMAP resource is used to map an inbound request from a web client to a servlet or JSP that is running in a JVM server.
URIMAP Scheme	The scheme for the HTTP request, HTTP with SSL (HTTPS) or without SSL (HTTP).
Authenticate	For USAGE(CLIENT), whether credentials (authentication information) are sent for outbound Web requests.
URIMAP Port	For USAGE(CLIENT), the port number used for the client connection. For USAGE(SERVER), the port number that is being used for the communication, even if PORT(NO) is specified on the URIMAP at define time.
URIMAP Host	For USAGE(CLIENT), the host name of the target URL to which the HTTP request is to be sent. For any other usage type, the host name on the incoming HTTP request that is used to select this URIMAP definition.
URIMAP IP Family	The address format of the address returned in URIMAP IP Resolved Address.
URIMAP IP Resolved Address	The IPv4 or IPv6 resolved address of the host.
URIMAP Path	For USAGE(CLIENT), the path of the target URL to which the HTTP request is to be sent. For any other usage type, the path on the incoming HTTP request that is used to select this URIMAP definition. The PATH might end in an asterisk, meaning that it is generic, and matches any path with characters that are the same up to but excluding the asterisk.

Table 215. URIMAP definitions: summary resource statistics (continued)

DFHSTUP name	Description
TCPIPSERVICE name	The TCPIPSERVICE resource to which this URIMAP definition applies. Only requests received using this TCPIPSERVICE resource are matched to this URIMAP definition. If no TCPIPSERVICE resource is specified, the URIMAP definition applies to all incoming HTTP requests.
WEBSERVICE name	The name of the WEBSERVICE resource definition for the web service that handles the incoming HTTP request.
PIPELINE name	The name of the PIPELINE resource definition for the web service that handles the incoming HTTP request.
ATOMSERVICE name	The name of the ATOMSERVICE resource definition for the Atom document.
Templatename	The name of a CICS document template, with the contents that are returned as the HTTP response.
zFS File	The name of a file in the z/OS UNIX System Services file system, with the contents that are returned as the HTTP response.
Analyzer	Whether the analyzer associated with the TCPIPSERVICE definition is called to process the request.
Converter	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM.
Transaction ID	The name of the alias transaction that processes the incoming HTTP request.
Program name	The name of the application program that processes the incoming HTTP request.
Redirection type	Whether matching requests will be redirected, on a temporary or permanent basis.
Location for redirection	An alternative URL to which the Web client is redirected, if redirection is specified.
URIMAP reference count	Number of times this URIMAP definition was referenced.

Table 215. URIMAP definitions: summary resource statistics (continued)

DFHSTUP name	Description
Disabled	Number of times that this URIMAP host and path were matched, but the URIMAP definition was disabled.
Redirected	Number of times that this URIMAP host and path were matched and the number of times that the request was redirected.
Time out for pooled sockets	The time after which CICS discards pooled client HTTP connections created using this URIMAP resource if they are not reused.
Peak number of pooled sockets	Peak number of open client HTTP connections held in the pool for reuse.
Number of reclaimed sockets	Number of pooled connections that were closed in the pool by CICS because the CICS region had reached the MAXSOCKETS limit.
Number of timed out sockets	Number of pooled connections that were closed in the pool by CICS because they reached their timeout value without being reused.

User domain statistics

These statistics are not available online, and are mapped by the DFHUSGDS DSECT.

Related concepts:

"Interpreting 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 expensive in both processor time and input/output operations.

Interpreting 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 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 user ID.
- The group ID.
- The applid, which is not necessarily the same for all the users in a region. The applid is shipped with the user ID across MRO links.

- The port of entry, which can be the netname, for users signed on at z/OS Communications Server terminals, or the console name, for users signed on at consoles. It is null for other terminal types and for users associated with nonterminal 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 tries to locate that instance in its user directory. If the user instance exists with the parameters described, that instance is reused. The **USGDRRC** parameter records how many times reuse occurs. However, if the user instance does not exist, it must be added, requiring a call of the security domain and the external security manager. The **USGDRNFC** parameter 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. The **USGTORC** parameter records how many times a user instance is reused while it was being timed out, and the **USGTOMRT** parameter 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. The **USGTOEC** parameter records how many times this happens.

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

High values of **USRDELAY** can affect the ability of your security administrator 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 might require you to specify **USRDELAY=0**, which 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 that they are running has ended, so that their user control blocks must be reconstructed frequently. This reconstruction results in poor performance.

If you specify a low value for the **USRDELAY** system initialization parameter to ensure that CICS detects changes to RACF profiles promptly, you might want to increase this value, because for z/OS 1.11 and later, CICS is notified immediately if RACF profile changes occur. The primary impact of a high **USRDELAY** value is that the amount of storage used for RACF control blocks is increased.

User domain: Global statistics

Table 216. 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 reused. Reset characteristic: reset to zero
Timeout reuse count	USGTORC	the number of times a user instance is reused from the timeout queue.. 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 user instance 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.

Table 217. User domain: Summary global statistics

DFHSTUP name	Description
Average timeout reuse time	is the average time user instances remain on the timeout queue until they are reused.
Timeout reuse count	is the number of times a user instance is reused from the timeout queue.
Timeout expiry count	is the number of times a user instance remains on the timeout queue for a full USRDELAY interval without being reused, and is consequently deleted.
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 does not already exist in the directory.

SNA statistics

You can retrieve statistics for z/OS Communications Server by using the **EXEC CICS COLLECT STATISTICS VTAM** system command. They are mapped by the DFHA03DS DSECT.

Note: VTAM is now z/OS Communications Server.

Related concepts:

"Interpreting z/OS Communications Server statistics"
Notes on how to understand the statistice returned by the **EXEC CICS COLLECT STATISTICS VTAM** system command.

Related reference:

"Program Autoinstall report" on page 862
The Program Autoinstall report shows information and statistics about the status of program autoinstall, catalog program definitions, and the number of autoinstalls that were attempted, rejected, and failed.

Related information:

EXTRACT STATISTICS in Reference > System programming

Interpreting z/OS Communications Server statistics

Notes on how to understand the statistice returned by the **EXEC CICS COLLECT STATISTICS VTAM** system command.

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. The z/OS Communications Server 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 the Communications Server was required to queue incoming messages in subpool 229 when no receive-any was available to accept the input. You should limit this Communications Server 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 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 the Communications Server 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 in this example, 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 means that a CICS request for service from the Communications Server was rejected with a Communications Server sense code indicating that the Communications Server was unable to acquire the storage required to service the request. The Communications Server does not give any further information to CICS, such as what storage it was unable to acquire.

Note: VTAM is now the z/OS Communications Server.

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 the Communications Server is short on storage in its own region and increase Communications Server 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, the Communications Server 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 the Communications Server 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.

z/OS Communications Server: Global statistics

Table 218. z/OS Communications Server: Global statistics

DFHSTUP name	Field name	Description
Times at RPL maximum	A03RPLXT	is the number of times the peak RPLs posted value (A03RPLX) was reached. Reset characteristic: reset to zero

Table 218. z/OS Communications Server: Global statistics (continued)

DFHSTUP name	Field name	Description
Peak RPLs posted	A03RPLX	is the maximum number of receive-any request parameter lists (RPLs) that are posted by the Communications Server 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 Communications Server SYNAD exit in the CICS terminal control program each time the Communications Server indicates that there is a temporary Communications Server storage problem. Reset characteristic: reset to zero
Dynamic opens count	A03DOC	is the number of times the Communications Server access method control block (ACB) was opened through the control terminal. If the Communications Server 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) • Communications Server SNA LUs. 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) • Communications Server SNA LUs. 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 Communications Server sessions that persisted. Reset characteristic: reset to current value.

Table 218. z/OS Communications Server: Global statistics (continued)

DFHSTUP name	Field name	Description
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.

z/OS Communications Server: Summary global statistics

Summary statistics are not available online.

Table 219. z/OS Communications Server: Summary global statistics

DFHSTUP name	Description
Times at RPL maximum	is the total number of times the peak RPLs posted value was reached.
Peak RPLs posted	is the peak number of receive-any request parameter lists (RPLs) that are posted by the Communications Server on any one dispatch of terminal control.
Short on storage count	is a counter that is incremented in the Communications Server SYNAD exit in the CICS terminal control program each time the Communications Server indicates that there is a temporary Communications Server storage problem.
Dynamic opens count	is the total number of times that the Communications Server access method control block (ACB) was opened through the control terminal. If the Communications Server is started before CICS and stays active for the whole CICS run, this value is 0.
Average 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 Communications Server sessions that persisted.
PS opndst count	is the total number of persisting sessions that were successfully restored.

Table 219. z/OS Communications Server: Summary global statistics (continued)

DFHSTUP name	Description
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.

Web service statistics

Web services support in CICS enables CICS applications to act in the role of both web service provider and web service requester, where the services are defined by using web services description language (WSDL).

WEBSERVICE resource definitions are used to define aspects of the runtime environment for CICS application programs deployed in a web services setting. Statistics are provided for each WEBSERVICE resource definition, and a total use count for all WEBSERVICE definitions is also available.

For information about the web services report, see "Web Services report" on page 926.

Web services: Resource statistics

You can retrieve web services resource statistics by using the **EXEC CICS EXTRACT STATISTICS WEBSERVICE RESID** system command. They are mapped by the DFHPIWDS DSECT.

The resource information gives details of various attribute settings of each WEBSERVICE resource definition. A total use count for all WEBSERVICE definitions is also available.

Table 220. Web Services: resource statistics

DFHSTUP name	Field name	Description
WEBSERVICE Name	PIW_WEBSERVICE_NAME	The name of the WEBSERVICE resource definition. Reset characteristic: not reset
PIPELINE name	PIW_PIPELINE_NAME	The name of the PIPELINE resource that contains this WEBSERVICE resource. Reset characteristic: not reset
URIMAP name	PIW_URIMAP_NAME	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this WEBSERVICE resource definition. Reset characteristic: not reset

Table 220. Web Services: resource statistics (continued)

DFHSTUP name	Field name	Description
Web service description (WSDL)	PIW_WSDL_FILE	The file name of the Web service description (WSDL) file associated with the WEBSERVICE resource. Reset characteristic: not reset
Archive file	PIW_ARCHIVE_FILE	The file name of the archive file containing one or more web service description (WSDL) files associated with the WEBSERVICE resource. Reset characteristic: not reset
Web service binding file	PIW_WSBIND_FILE	The file name of the Web service binding file associated with the WEBSERVICE resource. Reset characteristic: not reset
Web service WSDL binding	PIW_WSDL_BINDING	The WSDL binding represented by the WEBSERVICE resource. This binding is one of (potentially) many that appear in the WSDL file. Reset characteristic: not reset
Endpoint	PIW_ENDPOINT_URI	The URI specifying the location on the network (or endpoint) of the web service, as defined in the web service description. Reset characteristic: not reset
Validation	PIW_MSG_VALIDATION	Indicates whether full validation of SOAP messages against the corresponding schema in the web service description is specified. Reset characteristic: not reset
Program interface	PIW_PROGRAM_INTERFACE	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel. Reset characteristic: not reset
Program name	PIW_WEBSERVICE_PROGRAM	The name of the target application program. Reset characteristic: not reset

Table 220. Web Services: resource statistics (continued)

DFHSTUP name	Field name	Description
Container	PIW_CONTAINER_NAME	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top-level data. Reset characteristic: not reset
WEBSERVICE use count	PIW_WEBSERVICE_USE_COUNT	The number of times this WEBSERVICE resource definition was used to process a message. Reset characteristic: reset to zero
Not in DFHSTUP report	PIW_WEBSERVICE_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	PIW_WEBSERVICE_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	PIW_WEBSERVICE_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	PIW_WEBSERVICE_CHANGE_AGENT	Identifies the agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	PIW_WEBSERVICE_INSTALL_AGENT	Identifies the agent that installed the resourcee. Reset characteristic: not reset
Not in DFHSTUP report	PIW_WEBSERVICE_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset

Table 220. Web Services: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	PIW_WEBSERVICE_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

WEBSERVICE totals: The resource statistics also include a total WEBSERVICE use count, which shows the total number of times a WEBSERVICE resource definition was used to process a message.

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

Web services: Summary resource statistics

The resource information gives details of various attribute settings of each WEBSERVICE resource definition.

Summary statistics are not available online.

Table 221. Web services: Summary resource statistics

DFHSTUP name	Description
WEBSERVICE name	The name of the WEBSERVICE resource definition.
PIPELINE name	The name of the PIPELINE resource that contains this WEBSERVICE resource.
URIMAP name	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this WEBSERVICE.
Web service description (WSDL)	The file name of the web service description (WSDL) file associated with the WEBSERVICE resource.
Archive file	The file name of the archive file containing one or more web service description (WSDL) files associated with the WEBSERVICE resource.
Web service binding file	The file name of the web service binding file associated with the WEBSERVICE resource.

Table 221. Web services: Summary resource statistics (continued)

DFHSTUP name	Description
Web service WSDL binding	The WSDL binding represented by the WEBSERVICE. This binding is one of (potentially) many that appear in the WSDL file.
Endpoint	The URI specifying the location on the network (or endpoint) of the web service, as defined in the web service description.
Validation	Indicates whether full validation of SOAP messages against the corresponding schema in the web service description is specified.
Program interface	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel.
Program name	The name of the target application program.
Container	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top level data.
WEBSERVICE use count	The number of times this WEBSERVICE resource definition was used to process a message.

WEBSERVICE Totals:

The summary statistics also include a total WEBSERVICE use count, which shows the total number of times a WEBSERVICE resource definition was used to process a message.

WebSphere MQ Connection statistics

Related reference:

"WebSphere MQ Connection report" on page 927
The WebSphere MQ Connection report is produced using the **EXEC CICS EXTRACT STATISTICS MQCONN** command. The statistics data is mapped by the **DFHMQGDS DSECT**.

WebSphere MQ Connection statistics

You can retrieve WebSphere MQ Connection statistics by using the **EXEC CICS EXTRACT STATISTICS MQCONN** system command. They are mapped by the DFHMQGDS DSECT.

Summary global statistics for the WebSphere MQ Connection are also available in the *WebSphere MQ Connection: Summary global statistics* report. Summary statistics are not available online. The summary global statistics for the WebSphere MQ Connection include the same fields as the global statistics, except for the fields relating to the current connection status and tasks, which are not present in the summary statistics.

Table 222. WebSphere MQ Connection: Global statistics

DFHSTUP name	Field name	Description
MQCONN name	MQG_MQCONN_NAME	The name of the installed MQCONN definition for the CICS region, which defines the attributes of the connection between CICS and WebSphere MQ. Reset characteristic: not reset
WebSphere MQ Connect Date / Time	MQG_CONNECT_TIME_LOCAL	The date and time when the most recent connection between CICS and WebSphere MQ was started. In the summary statistics, this field is not present; instead, a field Total WebSphere MQ Connection Time shows the total time for which CICS was connected to WebSphere MQ. Reset characteristic: not reset
WebSphere MQ Connection Status	MQG_CONNECTION_STATUS	The status of the connection between CICS and WebSphere MQ: **C** Connected **N** Not connected Reset characteristic: not reset In the summary statistics, this field is not present.
WebSphere MQ Disconnect Date / Time	MQG_ DISCONNECT_TIME_LOCAL	The date and time when the most recent connection between CICS and WebSphere MQ ended. If CICS is currently connected to WebSphere MQ, this field is blank. In the summary statistics, this field is not present. Reset characteristic: not reset
Mqname	MQG_MQNAME	The name of the WebSphere MQ queue manager or queue-sharing group that is specified in the MQNAME attribute of the installed MQCONN definition for the CICS region. CICS uses this as the default for the connection. Reset characteristic: not reset
WebSphere MQ Queue Manager name	MQG_QMGR_NAME	The name of the WebSphere MQ queue manager to which CICS is currently connected. If CICS is not connected to WebSphere MQ, this field is blank. Reset characteristic: not reset

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Resync Group member	MQG_RESYNCMEMBER	Shows whether the MQCONN definition for the CICS region specifies resynchronization if there are indoubt units of work when CICS reconnects to WebSphere MQ: **YES** CICS connects to the same queue manager, waiting, if necessary, until the queue manager becomes active. **NO** CICS makes one attempt to connect to the same queue manager. If that attempt fails, CICS connects to any member of the queue-sharing group. **GROUPRESYNC** CICS connects to any member of the queue-sharing group. The queue manager is chosen by WebSphere MQ and it asks CICS to resolve indoubt units of work on behalf of all eligible queue managers in the queue-sharing group. This function is called group unit of recovery. Reset characteristic: not reset
WebSphere MQ Release	MQG_MQ_RELEASE	The release of WebSphere MQ that is connected to CICS.
Initiation Queue name	MQG_INITIATION_QUEUE	The name of the default initiation queue for the connection between CICS and WebSphere MQ. Reset characteristic: not reset
Number of current tasks	MQG_TTasks	The number of current tasks that have issued an MQI call. In the summary statistics, this field is not present. Reset characteristic: not reset
Number of futile attempts	MQG_TFutilAtt	A count of the number of MQI calls made while the connection status is “not connected”. This is reset to zero when the connection is established. Reset characteristic: reset to zero
Total number of API calls	MQG_TApi	The total number of MQI calls since the connection was made. Reset characteristic: reset to zero
Number of API calls completed OK	MQG_TApiOk	The total number of calls that have completed successfully. Reset characteristic: reset to zero

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of OPEN requests	MQG_TOPEN	The number of MQOPEN calls issued. Reset characteristic: reset to zero
Number of CLOSE requests	MQG_TCLOSE	The number of MQCLOSE calls issued. Reset characteristic: reset to zero
Number of GET requests	MQG_TGET	The number of MQGET calls issued. Reset characteristic: reset to zero
Number of GETWAIT requests	MQG_TGETWAIT	The number of MQGET calls issued with the MQGMO_WAIT option. Reset characteristic: reset to zero
Number of GETWAITs that waited	MQG_TWaitMsg	The number of MQGET calls issued with the MQGMO_WAIT option that waited for a message. Reset characteristic: reset to zero
Number of PUT requests	MQG_TPUT	The number of MQPUT calls issued. Reset characteristic: reset to zero
Number of PUT1 requests	MQG_TPUT1	The number of MQPUT1 calls issued. Reset characteristic: reset to zero
Number of INQ requests	MQG_TINQ	The number of MQINQ calls issued. Reset characteristic: reset to zero
Number of SET requests	MQG_TSET	The number of MQSET calls issued. Reset characteristic: reset to zero
Number of internal MQ calls	MQG_TCall	The total number of flows to WebSphere MQ on the connection. Reset characteristic: reset to zero
Number that completed synchronously	MQG_TCallSyncComp	The total number of calls completed synchronously. Reset characteristic: reset to zero

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Number that needed I/O	MQG_TCallIO	The total number of calls that needed I/O. Reset characteristic: reset to zero
Number of calls with TCB switch	MQG_TSubtasked	The number of API calls with a TCB switch. Reset characteristic: reset to zero
Number of indoubt units of work	MQG_IndoubtUOW	The number of indoubt UOWs at adapter startup. Reset characteristic: reset to zero
Number of unresolved units of work	MQG_UnResolvedUOW	The number of UOWs that were in doubt at adapter startup, and that have not been resolved because of a CICS cold start. Reset characteristic: reset to zero
Number of resolved committed UOWs	MQG_ResolveComm	The number of UOWs that were in doubt at adapter startup that have now been resolved by committing. Reset characteristic: reset to zero
Number of resolved backout UOWs	MQG_ResolveBack	The number of UOWs that were in doubt at adapter startup that have now been resolved by backing out. Reset characteristic: reset to zero
Number of Backout UOWs	MQG_TBackUOW	The total number of backed out UOWs. Reset characteristic: reset to zero
Number of Committed UOWs	MQG_TCommUOW	The total number of committed UOWs. Reset characteristic: reset to zero
Number of tasks	MQG_TTaskend	The total number of tasks. Reset characteristic: reset to zero
Number of Single Phase Commits	MQG_TSPComm	The total number of single-phase commits. Reset characteristic: reset to zero
Number of Two Phase Commits	MQG_T2PComm	The total number of two-phase commits. Reset characteristic: reset to zero

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of CB requests	MQG_TCB	The number of MQCB calls issued. Reset characteristic: reset to zero
Number of msgs consumed	MQG_TCONSUME	The number of messages passed to callback routines. Reset characteristic: reset to zero
Number of CTL requests	MQG_TCTL	The number of MQCTL calls issued. Reset characteristic: reset to zero
Number of SUB requests	MQG_TSUB	The number of MQSUB calls issued. Reset characteristic: reset to zero
Number of SUBRQ requests	MQG_TSUBRQ	The number of MQSUBRQ calls issued. Reset characteristic: reset to zero
Number of STAT requests	MQG_TSTAT	The number of MQSTAT calls issued. Reset characteristic: reset to zero
Number of CRTMH requests	MQG_TCRTMH	The number of MQCRTMH calls issued. Reset characteristic: reset to zero
Number of DLTMH requests	MQG_TDLTMH	The number of MQDLTMH calls issued. Reset characteristic: reset to zero
Number of SETMP requests	MQG_TSETMP	The number of MQSETMP calls issued. Reset characteristic: reset to zero
Number of INQMP requests	MQG_TINQMP	The number of MQINQMP calls issued. Reset characteristic: reset to zero
Number of DLTMP requests	MQG_TDLTMP	The number of MQDLTMP calls issued. Reset characteristic: reset to zero
Number of MHBUF requests	MQG_TMHBUF	The number of MQMHBUF calls issued. Reset characteristic: reset to zero

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Number of BUFMH requests	MQG_TBUFMH	The number of MQBUFMH calls issued. Reset characteristic: reset to zero
Not in DFHSTUP report	MQG_Connect_time_gmt	The Greenwich mean time (GMT) when CICS connected to WebSphere MQ. The DFHSTUP report expresses this time as `hh:mm:ss`; however, the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
Not in DFHSTUP report	MQG_Disconnect_time_gmt	The Greenwich mean time (GMT) when CICS disconnected to WebSphere MQ. The DFHSTUP report expresses this time as `hh:mm:ss`; however, the DSECT field contains the time as a GMT store clock (STCK) value. Reset characteristic: not reset
Not in DFHSTUP report	MQG_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	MQG_CHANGE_TIME	The time stamp (STCK) in local time of CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	MQG_CHANGE_USERID	The user ID that ran the change agent. Reset characteristic: not reset
Not in DFHSTUP report	MQG_CHANGE_AGENT	The agent that made the last change. Reset characteristic: not reset
Not in DFHSTUP report	MQG_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset
Not in DFHSTUP report	MQG_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset

Table 222. WebSphere MQ Connection: Global statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	MQG_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

XMLTRANSFORM statistics

The markup language (ML) domain collects statistics for XMLTRANSFORM resources, which define the XML binding and schema to transform application data to XML and vice versa.

CICS dynamically creates XMLTRANSFORM resources for you when you install BUNDLE or ATOMSERVICE resources.

Related reference:

"XMLTRANSFORMs report" on page 930
The XMLTRANSFORMs report shows information and statistics about XMLTRANSFORM resources. The XMLTRANSFORM resource defines where the XML binding is located on z/OS UNIX and its status. CICS dynamically creates an XMLTRANSFORM resource when you install a BUNDLE or ATOMSERVICE resource.

XMLTRANSFORM: resource statistics

You can retrieve XMLTRANSFORM resource statistics by using the **EXEC CICS EXTRACT STATISTICS XMLTRANSFORM** system command. They are mapped by the DFHMLRDS DSECT.

Table 223. XMLTRANSFORM: resource statistics

DFHSTUP name	Field name	Description
XMLTRANSFORM name	MLR_XMLTRANSFORM_NAME	The name of the XMLTRANSFORM resource. Reset characteristic: not reset

Table 223. XMLTRANSFORM: resource statistics (continued)

DFHSTUP name	Field name	Description
XML binding file	MLR_XSDBIND_FILE	The name and location of the XML binding in z/OS UNIX. Reset characteristic: not reset
XML schema file	MLR_XMLSCHEMA_FILE	The name and location of the XML schema in z/OS UNIX. Reset characteristic: not reset
Validation	MLR_MSG_VALIDATION	The status of XML validation. Reset characteristic: not reset
XMLTRANSFORM use count	MLR_XMLTRNFM_USE_COUNT	The number of times that the XML binding has been used for data transformation. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_DEFINE_SOURCE	The source of the resource definition. Its value depends on the change agent. For more information, see Summary of the resource signature field values in Product overview. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_CHANGE_TIME	The time stamp (STCK) in local time of the CSD record change. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_CHANGE_USERID	The user ID that ran the CHANGE_AGENT. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_CHANGE_AGENT	The agent that was used to make the last change. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_INSTALL_AGENT	The agent that installed the resource. Reset characteristic: not reset

Table 223. XMLTRANSFORM: resource statistics (continued)

DFHSTUP name	Field name	Description
Not in DFHSTUP report	MLR_XMLTRNFM_INSTALL_TIME	The time stamp (STCK) in local time when the resource was installed. Reset characteristic: not reset
Not in DFHSTUP report	MLR_XMLTRNFM_INSTALL_USERID	The user ID that installed the resource. Reset characteristic: not reset

The resource statistics fields for the resource signature

The resource signature captures details about when the resource is defined, installed, and last changed. The resource statistics field names for the resource signature end in CHANGE_AGENT, CHANGE_TIME, CHANGE_USERID, DEFINE_SOURCE, INSTALL_AGENT, INSTALL_TIME, and INSTALL_USERID. For detailed information about the content of the resource signature fields, see Summary of the resource signature field values in Product overview.

Related information:

EXTRACT STATISTICS in Reference > System programming

XMLTRANSFORM: Summary resource statistics

Summary statistics are not available online.

Table 224. XMLTRANSFORM: Summary resource statistics

DFHSTUP name	Description
XMLTRANSFORM name	The name of the XMLTRANSFORM resource.
XML binding file	The name and location of the XML binding in z/OS UNIX.
XML schema file	The name and location of the XML schema in z/OS UNIX.
Validation	The status of XML validation.
XMLTRANSFORM use count	The number of times that the XML binding has been used for data transformation.

Chapter 33. DFH0STAT reports

The sample statistics program DFH0STAT can produce reports about the statistics listed here. You can select the required statistics reports using the CICS Statistics Print Report Selection panels.

The heading of each report includes the generic APPLID, SYSID, job name, date, time, and the CICS version and release information.

ATOMSERVICEs report

The ATOMSERVICEs report shows information and statistics about ATOMSERVICE resource definitions, which define Atom feeds. This report is produced using a combination of **EXEC CICS INQUIRE ATOMSERVICE** and **EXEC CICS EXTRACT STATISTICS ATOMSERVICE** commands.

The statistics data is mapped by the DFHW2RDS DSECT.

Table 225. Fields in the ATOMSERVICEs report

Field Heading	Description
ATOMSERVICE Name	The name of the ATOMSERVICE resource definition. Source field: **EXEC CICS INQUIRE ATOMSERVICE**
ATOMSERVICE Enable Status	Whether the ATOMSERVICE definition is enabled or disabled. Source field: **EXEC CICS INQUIRE ATOMSERVICE() ENABLESTATUS**
Atom document type	The type of Atom document that is returned for this ATOMSERVICE resource definition. **Category** An Atom category document, which lists the categories for entries in a collection. **Collection** An Atom collection document, which contains a group of entry documents that can be edited. **Feed** An Atom feed document, which describes the metadata for a feed, and contains entry documents that provide data for the feed. **Service** An Atom service document, which provides information about the collections that are available on the server. Source field: **EXEC CICS INQUIRE ATOMSERVICE() ATOMTYPE**
Atom configuration file	The name of the Atom configuration file containing the XML for the Atom document. Source field: **EXEC CICS INQUIRE ATOMSERVICE() CONFIGFILE**
Atom binding file	The name of the Atom binding file for the resource used for the Atom feed. Source field: **EXEC CICS INQUIRE ATOMSERVICE() BINDFILE**

Table 225. Fields in the ATOMSERVICEs report (continued)

Field Heading	Description
Resource type for Atom feed	The type of resource that provides the data for this Atom feed. **File** A CICS file. **Program** A service routine, which is a CICS application program written to supply content for Atom entries. **Tsqueue** A temporary storage queue. Source field: **EXEC CICS INQUIRE ATOMSERVICE() RESOURCETYPE**
Resource name for Atom feed	The name of the resource definition for the CICS resource that provides the data for this Atom feed or collection. Source field: **EXEC CICS INQUIRE FILE() DSNAME**
Dataset name	For resources of type File only, the name of the data set containing the file that provides the data for this Atom feed or collection. Source field: **EXEC CICS INQUIRE ATOMSERVICE() RESOURCENAME**
ATOMSERVICE reference count	The number of times this ATOMSERVICE resource definition was referenced. Source field: W2R-ATOMSERV-REF-COUNT
Disabled	The number of times this ATOMSERVICE resource definition was referenced, but the resource definition was disabled. Source field: W2R-ATOMSERV-REF-DISABLED
POST requests to the feed URL	The number of HTTP POST requests to add a new Atom entry to this Atom feed or collection. Source field: W2R-ATOMSERV-POST-FEED-CNT
GET requests to the feed URL	The number of HTTP GET requests to obtain a group of entries from this Atom feed or collection. Source field: W2R-ATOMSERV-GET-FEED-CNT
GET requests to the entry URL	The number of HTTP GET requests to obtain an individual Atom entry from this Atom feed or collection. Source field: W2R-ATOMSERV-GET-ENTRY-CNT
PUT requests to the entry URL	The number of HTTP PUT requests to edit an Atom entry in this Atom feed or collection. Source field: W2R-ATOMSERV-PUT-ENTRY-CNT
DELETE requests to the entry URL	The number of HTTP DELETE requests to delete an individual Atom entry from this Atom feed or collection. Source field: W2R-ATOMSERV-DEL-ENTRY-CNT

Bundles Report

The Bundles Report shows information and statistics about BUNDLE resources. The BUNDLE resource defines where a CICS bundle is deployed on z/OS UNIX and its status.

This report is produced using a combination of **EXEC CICS INQUIRE BUNDLE** and **EXEC CICS EXTRACT STATISTICS BUNDLE** commands. The statistics data is mapped by the DFHRLRDS DSECT.

Table 226. Fields in the Bundles report

Field Heading	Description
BUNDLE Name	The name of the BUNDLE resource. Source field: **EXEC CICS INQUIRE BUNDLE**
BUNDLE Enable Status	The status of the BUNDLE resource, either enabled or disabled. Source field: **EXEC CICS INQUIRE BUNDLE () ENABLESTATUS**
BUNDLE Directory	The location of the CICS bundle in z/OS UNIX. Source field: **EXEC CICS INQUIRE BUNDLE () BUNDLEDIR**
BUNDLE Scope Name	The scope of the bundle, as specified in the BASESCOPE attribute on the BUNDLE resource. For a bundle that is part of an application or platform, the scope is a URI that lists the platform, application, and application version. Source field: **EXEC CICS INQUIRE BUNDLE () BASESCOPE**
BUNDLEPART count	The number of imports, exports, entry points, policy scopes, and define statements that are defined in the bundle manifest. Source field: **EXEC CICS INQUIRE BUNDLE () PARTCOUNT**
Target enabled definitions	The total number of resources, entry points, and policy scopes that the bundle creates when enabled. Source field: **EXEC CICS INQUIRE BUNDLE () TARGETCOUNT**
Current enabled definitions	The number of resources, entry points, and policy scopes that were created by the bundle and are currently enabled in the CICS region. Source field: **EXEC CICS INQUIRE BUNDLE () ENABLEDCOUNT**

Connections and Modenames report

The Connections and Modenames report is produced using a combination of the **EXEC CICS INQUIRE CONNECTION**, **EXEC CICS INQUIRE MODENAME** and **EXEC CICS COLLECT STATISTICS CONNECTION** commands. The statistics data is mapped by the DFHA14DS DSECT.

Table 227. Fields in the Connections report

Field Heading	Description
Connection Name/Netname	The connection name (sysid) and the network name (applid) for the connection. Source field: **EXEC CICS INQUIRE CONNECTION() NETNAME()**
Access Method/Protocol	The communication access method and protocol used for the connection. Source field: **EXEC CICS INQUIRE CONNECTION() ACCESSMETHOD(cvda) PROTOCOL(cvda)**
Autoinstalled Connection Create Time	The local time at which this connection was autoinstalled. This field applies to APPC connections only. Source field: A14AICT
Peak Contention Losers	The peak number of contention loser sessions that were in use. Source field: A14E1HWM
ATIs satisfied by Losers	The number of queued allocate requests that have been satisfied by contention loser sessions. Source field: A14ES1
Receive Session Count	The number of receive sessions for this connection. (MRO and LU6.1 connections only) Source field: **EXEC CICS INQUIRE CONNECTION() RECEIVECOUNT()**
Send Session Count	The number of send sessions for this connection. (MRO and LU6.1 connections only) Source field: **EXEC CICS INQUIRE CONNECTION() SENDCOUNT()**
Peak Contention Winners	The peak number of contention winner sessions that were in use. Source field: A14E2HWM
ATIs satisfied by Winners	The number of queued allocate requests that have been satisfied by contention winner sessions. Source field: A14ES2
Current AIDs in chain	The current number of automatic initiate descriptors (AIDs) in the AID chain. Source field: A14EALL
Generic AIDs in chain	The current number of automatic initiate descriptors (AIDs) that are waiting for a session to become available to satisfy the allocate request. Source field: A14ESALL
Total number of Bids sent	The total number of bids sent. Source field: A14ESBID
Current Bids in progress	The current number of bids in progress. Source field: A14EBID
Peak Bids in progress	The peak number of bids that were in progress. Source field: A14EBHWM
Total Allocates	The total number of allocates for this connection. Source field: A14ESTAS
Allocates per second	The number of allocates issued per second for this connection. Source field: A14ESTAS / Elapsed seconds since reset

Table 227. Fields in the Connections report (continued)

Field Heading	Description
Allocates Queued	The current number of allocate requests queued for this connection. Source field: A14ESTAQ
Peak Allocates Queued	The peak number of allocate requests queued for this connection. Source field: A14ESTAM
Allocate Max Queue Time	The MAXQTIME value specified for this connection. Source field: A14EMXQT
Allocate Queue Limit	The last value encountered for the QUEUELIMIT parameter specified on the CONNECTION definition. When set, if this value is reached, then allocates are rejected. Source field: A14EALIM
Allocates Failed - Link	The number of allocate requests that failed due to the connection being released, out of service, or with a closed mode group. Source field: A14ESTAF
Allocates Failed - Other	The number of allocate requests that failed due to a session not being currently available for use. Source field: A14ESTAO
Allocates Rejected - Queue Limit	The number of allocate requests that were rejected due to the QUEUELIMIT value being reached. Source field: A14EALRJ
Max Queue Time - Allocate Purge	The number of times the allocate request queue has been purged due to the MAXQTIME value being reached. Source field: A14EQPCT
Allocates Purged - Max Queue Time	The total number of allocate requests purged due to the queueing time exceeding the MAXQTIME value. Source field: A14EMQPC
Transaction Routing - Total	The total number of transaction routing requests sent across the connection. Source field: A14ESTTC
Transaction Routing - Channel	The number of transaction routing requests sent across the connection, with channels. This is a subset of Transaction Routing - Total. Source field: A14ESTTC-CHANNEL
Allocates Rejected - XZIQUE	The number of allocate requests that were rejected by a XZIQUE global user exit. Source field: A14EZQRJ
XZIQUE - Allocate Purge	The number of times the allocate request queue has been purged by a XZIQUE global user exit. Source field: A14EZQPU
Allocates Purged - XZIQUE	The total number of allocate requests purged due to a XZIQUE global user exit requesting that the queued allocate requests should be purged. Source field: A14EZQPC

Table 227. Fields in the Connections report (continued)

Field Heading	Description
Function Shipping Requests: File Control	The number of file control requests function shipped across the connection. Source field: A14ESTFC
Function Shipping Requests: Interval Control - Total	The total number of interval control requests function shipped across the connection. Source field: A14ESTIC
Function Shipping Requests: Interval Control - Channel	The number of interval control requests, with channels, function shipped across the connection. This is a subset of Function Shipping Requests: Interval Control - Total. Source field: A14ESTIC-CHANNEL
Function Shipping Requests: Transient Data	The number of transient data requests function shipped across the connection. Source field: A14ESTTD
Function Shipping Requests: Temporary Storage	The number of temporary storage requests function shipped across the connection. Source field: A14ESTTS
Function Shipping Requests: Program Control - Total	The total number of program control requests function shipped across the connection. Source field: A14ESTPC
Function Shipping Requests: Program Control - Channel	The number of program control requests, with channels, function shipped across the connection. This is a subset of Function Shipping Requests: Program Control - Total. Source field: A14ESTPC-CHANNEL
Function Shipping Requests: Total	The total number of requests function shipped across the connection. Source field: A14ESTFC, A14ESTIC, A14ESTTD, A14ESTTS, A14ESTPC
Bytes Sent by Transaction Routing Requests	The number of bytes sent using channels, on transaction routing requests. This is the total amount of data sent using channels on the connection, including any control information. Source field: A14ESTTC-CHANNEL-SENT
Average Bytes Sent by Routing requests	The average number of bytes sent using channels, on transaction routing requests. Source field: A14ESTTC-CHANNEL-SENT / A14ESTTC-CHANNEL
Bytes Received by Transaction Routing Requests	The number of bytes received using channels, on transaction routing requests. This is the total amount of data received using channels on the connection, including any control information. Source field: A14ESTTC-CHANNEL-RCVD
Bytes Sent by Program Channel requests	The number of bytes sent on program control requests, with channels. This is the total amount of data sent on the connection for these requests, including any control information. Source field: A14ESTPC-CHANNEL-SENT
Average Bytes Sent by Channel request	The average number of bytes sent on program control requests, with channels. Source field: A14ESTPC-CHANNEL-SENT / A14ESTPC-CHANNEL

Table 227. Fields in the Connections report (continued)

Field Heading	Description
Bytes Received by Program Channel requests	The number of bytes received on program control requests, with channels. This is the total amount of data received on the connection for these requests, including any control information. Source field: A14ESTPC-CHANNEL-RCVD
Bytes Sent by Interval Channel requests	The number of bytes sent on interval control requests, with channels. This is the total amount of data sent on the connection for these requests, including any control information. Source field: A14ESTIC-CHANNEL-SENT
Average Bytes Sent by Channel request	The average number of bytes sent on interval control requests, with channels. Source field: A14ESTIC-CHANNEL-SENT / A14ESTIC-CHANNEL
Bytes Received by Interval Channel requests	The number of bytes received on interval control requests, with channels. This is the total amount of data received on the connection for these requests, including any control information. Source field: A14ESTIC-CHANNEL-RCVD

Table 228. Fields in the Modenames report

Field Heading	Description
Modename Connection Name	The name of the connection that owns this mode group entry. Source field: **EXEC CICS INQUIRE MODENAME() CONNECTION()**
Modename	The mode group name. Source field: **EXEC CICS INQUIRE MODENAME()**
Active Sessions	The number of sessions in this mode group currently in use. Source field: **EXEC CICS INQUIRE MODENAME() ACTIVE()**
Available Sessions	The current number of sessions in this mode group (bound). Source field: **EXEC CICS INQUIRE MODENAME() AVAILABLE()**
Maximum Sessions	The maximum number of sessions defined in this mode group. Source field: **EXEC CICS INQUIRE MODENAME() MAXIMUM()**
Maximum Contention Winners	The maximum number of sessions in this mode group that are defined to be contention winners. Source field: **EXEC CICS INQUIRE MODENAME() MAXWINNERS()**

Coupling Facility Data Table Pools report

The Coupling Facility Data Table Pools report shows information and statistics about Coupling Facility Data Table Pools, which contain one or more coupling facility data tables.

Table 229. Fields in the Coupling Facility Data Table Pools report

Field Heading	Description
Coupling Facility Data Table Pool	The name of the coupling facility data table pool. Source field: **EXEC CICS INQUIRE CFDTPOOL()**

Table 229. Fields in the Coupling Facility Data Table Pools report (continued)

Field Heading	Description
Connection Status	Indicates the connection status of the pool. Source field: **EXEC CICS INQUIRE CFDTPOOL() CONNSTATUS(cvda)**

Data Set Name report

The Data Set Name report is produced using the **EXEC CICS INQUIRE DSNAME** command.

Table 230. Fields in the Data Set Name report

Field Heading	Description
Data set name	The name of the data set. Source field: **EXEC CICS INQUIRE DSNAME()**
Access Method	The access method used with the data set. Source field: **EXEC CICS INQUIRE DSNAME() ACCESSMETHOD()**
Dsname Object	Indicates whether the object of the inquiry is a real data set containing records (a VSAM KSDS, ESDS, or RRDS, or an alternate index used directly) or a VSAM path definition that links an alternate index to its base cluster. BASE indicates a data set containing records. PATH indicates a VSAM path definition. A blank field in the report indicates either that the data set has not been opened by this CICS region, or that it is a BDAM data set. Source field: **EXEC CICS INQUIRE DSNAME() OBJECT()**
Dsname Validity	Indicates whether the data set name has been validated against the VSAM catalog by opening a file associated with the data set. Source field: **EXEC CICS INQUIRE DSNAME() VALIDITY()**
Dsname Availability	Indicates whether the data set is currently flagged, in this CICS region, as available or unavailable for use. Source field: **EXEC CICS INQUIRE DSNAME() AVAILABILITY()**
File Count	The number of installed file definitions that refer to this data set. Source field: **EXEC CICS INQUIRE DSNAME() FILECOUNT()**
Recovery Status	The recovery characteristics of the data set. Source field: **EXEC CICS INQUIRE DSNAME() RECOVSTATUS()**

Data Tables reports

The Data Tables Requests and Data Tables Storage reports are produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

Table 231. Fields in the Data Tables Requests report

Field Heading	Description
Filename	The name of the file. Source field: **EXEC CICS INQUIRE FILE()**

Table 231. Fields in the Data Tables Requests report (continued)

Field Heading	Description
Successful Reads	The number of attempts to retrieve records from the table. Source field: A17DTRDS
Records Not Found	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	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	The number of attempts to add records to the table as a result of WRITE requests. Source field: A17DTADS
Adds Rejected	The number of records CICS attempted to add to the table which were rejected by the global user exit. Source field: A17DTARJ
Adds Full	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	The number of attempts to update records in the table as a result of REWRITE requests. Source field: A17DTRWS
Delete Requests	The number of attempts to delete records from the table as a result of DELETE requests. Source field: A17DTDLS
Read Retries	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
Chng Resp/Lock Waits	For a CFDT that is using the locking model, records are locked when they are read for update. This count is the number of times it was necessary to WAIT for an already locked record. For a CFDT that is using the contention model, records are not locked when they are read for update. If a subsequent rewrite or delete request finds that the record has already changed a CHANGED response is returned. This count is the number of times that a CHANGED response was issued. Source field: A17DTCON

Table 232. Fields in the Data Tables Storage report

Field Heading	Description
Filename	The name of the file. Source field: **EXEC CICS INQUIRE FILE()**
Type	The type of data table, coupling facility, CICS-maintained or user-maintained. Source field: **EXEC CICS INQUIRE FILE() TABLE(cvda)**

Table 232. Fields in the Data Tables Storage report (continued)

Field Heading	Description
Current Records	The current number of records in the data table. Source field: A17DTSIZ
Peak Records	The peak number of records in the data table. Source field: A17DTSHI
Total - Storage Allocated	The total amount of storage (kilobytes) in allocated for the data table. Source field: A17DTALT
Total - Storage In-Use	The total amount of storage (kilobytes) in use for the data table. Source field: A17DTUST
Entries - Storage Allocated	The total amount of storage (kilobytes) allocated for the record entry blocks. Source field: A17DTALE
Entries - Storage In-Use	The total amount of storage (kilobytes) in use for the record entry blocks. Source field: A17DTUSE
Index - Storage Allocated	The total amount of storage (kilobytes) allocated for the index. Source field: A17DTALI
Index - Storage In-Use	The total amount of storage (kilobytes) in use for the index. Source field: A17DTUSI
Data - Storage Allocated	The total amount of storage (kilobytes) allocated for the record data. Source field: A17DTALD
Data - Storage In-Use	The total amount of storage (kilobytes) in use for the record data. Source field: A17DTUSD
Totals	Final total of the storage allocation for each storage column, for all the Data Tables listed in the report.

DB2 reports

There are two DB2 reports, the DB2 Connection report, and the DB2 Entries report.

DB2 Connection report

The DB2 Connection report shows information and statistics about DB2 Connection resource definitions, which define the connection between CICS and DB2 for a CICS region. The report also includes statistics about pool threads, DSNC commands, and tasks that wait for a TCB or pool thread.

This report is produced using a combination of the **EXEC CICS INQUIRE DB2CONN** and **EXEC CICS COLLECT STATISTICS DB2CONN** commands. The statistics data is mapped by the DFHD2GDS DSECT.

Table 233. Fields in the DB2 Connection report

Field Heading	Description
DB2 Connection Name	The name of the installed DB2CONN. Source field: D2G-DB2CONN-NAME

Table 233. Fields in the DB2 Connection report (continued)

Field Heading	Description
DB2 Group Id	The name of a data-sharing group of DB2 subsystems, specified in the installed DB2CONN definition. CICS connects to any active member of this group. Source field: D2G-DB2-GROUP-ID
Resync Group Member	If you are using group attach, specifies whether CICS attempts to resynchronize with the last connected DB2 data-sharing group member if outstanding units of work are being held. Source field: D2G-RESYNCMEMBER
DB2 Sysid	The name of the DB2 subsystem to which the CICS DB2 attachment is connected or will connect. If you are using group attach and the CICS DB2 attachment is connected or waiting to connect, this is the member of the data-sharing group of DB2 subsystems that has been chosen from the group. Source field: D2G-DB2-ID
DB2 Release	The version and release level of the DB2 subsystem to which CICS is currently connected. Source field: D2G-DB2-RELEASE
DB2 Connection Status	The current status of the CICS-DB2 Connection. Source field: **EXEC CICS INQUIRE DB2CONN CONNECTST**
DB2 Connect Date and Time	The date and time that the CICS connected to the DB2 subsystem. Source field: D2G-CONNECT-TIME-LOCAL
DB2 Connection Error	Specifies how CICS reports back to an application that issues an SQL request that CICS is not connected to DB2. Source field: **EXEC CICS INQUIRE DB2CONN CONNECTERROR**
DB2 Standby Mode	Specifies the action to be taken by the CICS-DB2 attachment if the DB2 subsystem is not active when an attempt to start the connection from CICS to DB2 is made. Source field: **EXEC CICS INQUIRE DB2CONN STANDBYMODE**
DB2 Pool Thread Plan Name	The name of the plan used for the pool. Source field: D2G-POOL-PLAN-NAME
DB2 Pool Thread Dynamic Plan Exit Name	The name of the dynamic plan exit used for pool threads. Source field: D2G-POOL-PLANEXIT-NAME
Dynamic Plan Exit Concurrency Status	Specifies whether the dynamic plan exit used for pool threads is defined as QUASIRENT or THREADSAFE. Source field: **EXEC CICS INQUIRE PROGRAM CONCURRENCY**
Pool Thread Authtype	The type of ID to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHTYPE
Command Thread Authtype	The type of ID to be used for security checking when using command threads. Source field: D2G-COMD-AUTHTYPE
Pool Thread Authid	The ID to be used for security checking when using pool threads. Source field: D2G-POOL-AUTHID

Table 233. Fields in the DB2 Connection report (continued)

Field Heading	Description
Command Thread Authid	The ID to be used for security checking when using command threads. Source field: D2G-COMD-AUTHID
Signid for Pool/Entry/Command Threads	The authorization ID to be used by the CICS-DB2 attachment when signing on to DB2 for pool threads and DB2 entry threads when **Pool Thread Authtype** is SIGNID and for command threads when **Command Thread Authtype** is SIGNID. Source field: **EXEC CICS INQUIRE DB2CONN SIGNID**
Create Thread Error	Specifies the action to be taken when a create thread error occurs. Source field: **EXEC CICS INQUIRE DB2CONN THREADERROR**
Message TD Queue 1	The name of the first transient data queue to which unsolicited messages from the CICS-DB2 attachment are sent. Source field: **EXEC CICS INQUIRE DB2CONN MSGQUEUE1**
Protected Thread Purge Cycle	The length of time (mm:ss) of the protected thread purge cycle. Source field: **EXEC CICS INQUIRE DB2CONN PURGECYCLEM** and **PURGECYCLES**
Message TD Queue 2	The name of the second transient data queue to which unsolicited messages from the CICS-DB2 attachment are sent. Source field: **EXEC CICS INQUIRE DB2CONN MSGQUEUE2**
Deadlock Resolution	The action to be taken for a transaction using a pool thread that has been selected by DB2 as victim of a deadlock resolution. Source field: **EXEC CICS INQUIRE DB2CONN DROLLBACK**
Message TD Queue 3	The name of the third transient data queue to which unsolicited messages from the CICS-DB2 attachment are sent. Source field: **EXEC CICS INQUIRE DB2CONN MSGQUEUE3**
Non-Terminal Intermediate Syncpoint	Specifies whether non-terminal transactions release threads for reuse at intermediate sync points. Source field: **EXEC CICS INQUIRE DB2CONN NONTERMREL**
Pool Thread Wait Setting	Specifies whether transactions should wait for a pool thread or be abended if the number of active pool threads reaches the pool thread limit. Source field: D2G-POOL-THREADWAIT
Statistics TD Queue	The name of the transient data queue for the CICS-DB2 attachment statistics produced when the CICS-DB2 attachment is shut down. Source field: **EXEC CICS INQUIRE DB2CONN STATSQUEUE**
Pool Thread Priority	The priority of the pool thread subtasks relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing 'Not Applicable'. Source field: D2G-POOL-PRIORITY
DB2 Accounting records by	Specifies the frequency of DB2 accounting records to be produced for transactions using pool threads. Source field: D2G-POOL-ACCOUNTREC

Table 233. Fields in the DB2 Connection report (continued)

Field Heading	Description
Current TCB Limit	The maximum number of TCBs that can be used by the CICS DB2 attachment facility. Source field: D2G-TCB-LIMIT
Thread Reuselimit	The number of times a thread can be reused before being terminated. Source field: D2G-REUSELIMIT
Current number of Connections	The current number of connections in use by the CICS DB2 attachment facility. Source field: D2G-TCB-CURRENT
Peak number of Connections	The peak number of connections used by the CICS DB2 attachment facility. Source field: D2G-TCB-HWM
Current number of free Connections	The number of free connections available for use with CICS open TCBs. Source field: D2G-TCB-FREE
Current number of tasks on TCB Readyq	The number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Source field: D2G-TCB-READYQ-CURRENT
Peak number of tasks on TCB Readyq	The peak number of CICS tasks queued waiting because the TCBLIMIT specified in the DB2CONN has been reached. Source field: D2G-TCB-READYQ-PEAK
Pool Thread Limit	The maximum number of pool threads allowed. Source field: D2G-POOL-THREAD-LIMIT
Number of Calls using Pool Threads	The number of SQL calls made using pool threads. Source field: D2G-POOL-CALLS
Current number of Pool Threads	The current number of active pool threads. Source field: D2G-POOL-THREAD-CURRENT
Number of Pool Thread Signons	The number of DB2 signons performed for pool threads. Source field: D2G-POOL-SIGNONS
Peak number of Pool Threads	The peak number of active pool threads. Source field: D2G-POOL-THREAD-HWM
Number of Pool Thread Partial Signons	The number of DB2 partial signons performed for pool threads. Source field: D2G-POOL-PARTIAL-SIGNONS
Number of Pool Thread Waits	The number of times all available threads in the pool were busy and a transaction had to wait for a thread to become available. This count includes transactions that overflow to the pool to acquire a thread and have to wait for a pool thread. Source field: D2G-POOL-THREAD-WAITS
Number of Pool Thread Commits	The number of two phase commits performed for units of work using pool threads. Source field: D2G-POOL-COMMITS
Number of Pool Thread Aborts	The number of units of work using pool threads that were rolled back. Source field: D2G-POOL-ABORTS

Table 233. Fields in the DB2 Connection report (continued)

Field Heading	Description
Current number of Pool Tasks	The current number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-CURRENT
Number of Pool Thread Single Phase	The number of units of work using pool threads that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2G-POOL-SINGLE-PHASE
Peak number of Pool Tasks	The peak number of CICS tasks using pool threads. Source field: D2G-POOL-TASK-HWM
Number of Pool Thread Reuses	The number of times CICS transactions using the pool were able to reuse an already created DB2 thread. This count includes transactions that overflow to the pool to acquire a thread and reuse an existing thread. Source field: D2G-POOL-THREAD-REUSE
Current Total number of Pool Tasks	The current total number of tasks that have used a pool thread. Source field: D2G-POOL-TASK-TOTAL + D2G-POOL-TASK-CURRENT
Number of Pool Thread Terminates	The number of terminate thread requests made to DB2 for pool threads. This includes pool threads used by transactions that overflow to the pool. Source field: D2G-POOL-THREAD-TERM
Current number of Tasks on Pool Readyq	The current number of CICS tasks waiting for a pool thread to become available. Source field: D2G-POOL-READYQ-CURRENT
Times reuselimit hit by a pool thread	The number of times the reuselimit has been reached by a pool thread. Source field: D2G_POOL_REUSELIMIT_COUNT
Peak number of Tasks on Pool Readyq	The peak number of CICS tasks that waited for a pool thread to become available. Source field: D2G-POOL-READYQ-HWM
Current number of DSNC Command threads	The current number of active command threads servicing DB2 commands issued using the DSNC transaction. Source field: D2G-COMD-THREAD-CURRENT
Number of DSNC Command Calls	The number of DB2 commands issued using the DSNC transaction. Source field: D2G-COMD-CALLS
Peak number of DSNC Command threads	The peak number of command threads servicing DSNC DB2 commands. Source field: D2G-COMD-THREAD-HWM
Number of DSNC Command Signons	The number of DB2 signons performed for DSNC DB2 commands. Source field: D2G-COMD-SIGNONS
DSNC Command Thread Limit	The maximum number of command threads allowed for DSNC DB2 commands. Source field: D2G-COMD-THREAD-LIMIT
Number of DSNC Command Thread Terminates	The number of terminate thread requests made to DB2 for command threads. Source field: D2G-COMD-THREAD-TERM

Table 233. Fields in the DB2 Connection report (continued)

Field Heading	Description
Number of DSNC Command Thread Overflows	The number of times a DSNC DB2 command resulted in a pool thread being used because of the active number of command threads exceeding the command thread limit. Source field: D2G-COMD-THREAD-OVERF

DB2 Entries report

The DB2 Entries Report is produced using a combination of the **EXEC CICS INQUIRE DB2ENTRY** and **EXEC CICS EXTRACT STATISTICS DB2ENTRY** commands. The statistics data is mapped by the **DFHD2RDS DSECT**.

Table 234. Fields in the DB2 Entries report

Field Heading	Description
DB2Entry Name	The name of the installed DB2ENTRY. Source field: EXEC CICS INQUIRE DB2ENTRY
DB2Entry Static Plan Name	The name of the plan to be used for this DB2ENTRY. Source field: D2R-PLAN-NAME
DB2Entry Dynamic Plan Exit Name	The name of the dynamic plan exit used by this DB2ENTRY. Source field: D2R-PLANEXIT-NAME
Dynamic Plan Exit Concurrency Status	Whether the dynamic plan exit used by this DB2ENTRY is defined as QUASIRENT, THREADSAFE, or REQUIRED. Source field: **EXEC CICS INQUIRE PROGRAM CONCURRENCY**
DB2Entry Status	The current enabled status of this DB2ENTRY. Source field: **EXEC CICS INQUIRE DB2ENTRY ENABLESTATUS**
DB2Entry Disabled Action	The action to be taken for new CICS tasks that attempt to use this DB2ENTRY when it is disabled or being disabled. Source field: **EXEC CICS INQUIRE DB2ENTRY DISABLEDACT**
DB2Entry Deadlock Resolution	The action to be taken for a transaction using a thread from this DB2ENTRY that has been selected by DB2 as a victim of deadlock resolution. Source field: **EXEC CICS INQUIRE DB2ENTRY DROLLBACK**
DB2Entry Authtype	The type of id to be used for security checking for threads of this DB2ENTRY. Source field: D2R-AUTHTYPE
DB2Entry Accounting records by	specifies the frequency of DB2 accounting records to be produced for transactions using this DB2ENTRY. Source field: D2R-ACCOUNTREC
DB2Entry Authid	The id to be used for security checking for threads of this DB2ENTRY. Source field: D2R-AUTHID
Number of Calls using DB2Entry	The number of SQL calls made using a thread from this DB2ENTRY. Source field: D2R-CALLS

Table 234. Fields in the DB2 Entries report (continued)

Field Heading	Description
DB2Entry Thread Wait Setting	specifies whether transactions should wait for a DB2ENTRY thread, be abended, or overflow to the pool should the number of active threads reach the thread limit for this DB2ENTRY. Source field: D2R-THREADWAIT
Number of DB2Entry Signons	The number of DB2 signons performed for threads of this DB2ENTRY. Source field: D2R-SIGNONS
Number of DB2Entry Partial Signons	The number of DB2 partial signons performed for threads of this DB2ENTRY. Source field: D2R-PARTIAL-SIGNONS
DB2Entry Thread Priority	The priority of the thread subtasks for this DB2ENTRY relative to the CICS main task (QR TCB). If CICS is connected to DB2 Version 6 or later, this field contains zero, representing "Not Applicable". Source field: D2R-PRIORITY
Number of DB2Entry Commits	The number of two phase commits performed for units of work using threads from this DB2ENTRY. Source field: D2R-COMMITS
DB2Entry Thread Limit	The maximum number of threads allowed for this DB2ENTRY. Source field: D2R-THREAD-LIMIT
Number of DB2Entry Aborts	The number of units of work using threads from this DB2ENTRY that were rolled back. Source field: D2R-ABORTS
Current number of DB2Entry Threads	The current number of active threads using this DB2ENTRY. Source field: D2R-THREAD-CURRENT
Number of DB2Entry Single Phase	The number of units of work using threads from this DB2ENTRY that used single-phase commit, either because they were read-only UOWs, or because DB2 was the only recoverable resource updated in the UOW. Source field: D2R-SINGLE-PHASE
Peak number of DB2Entry Threads	The peak number of active threads for this DB2ENTRY. Source field: D2R-THREAD-HWM
Number of DB2Entry Thread Reuses	The number of times CICS transactions using this DB2ENTRY were able to reuse an already created DB2 thread. Source field: D2R-THREAD-REUSE
Number of DB2Entry Thread Terminates	The number of terminate thread requests made for threads for this DB2ENTRY. Source field: D2R-THREAD-TERM
DB2Entry Protected Thread Limit	The maximum number of protected threads allowed for this DB2ENTRY. Source field: D2R-PTHREAD-LIMIT
Number of DB2Entry Thread Waits/Overflows	The number of times all available threads for this DB2ENTRY were busy and a transaction must wait for a thread to become available or overflow to the pool and use a pool thread. Source field: D2R-THREAD-WAIT-OR-OVERFL

Table 234. Fields in the DB2 Entries report (continued)

Field Heading	Description
Current number of DB2Entry Protected Threads	The current number of inactive threads of this DB2ENTRY that are protected. Source field: D2R-PTHREAD-CURRENT
Peak number of DB2Entry Protected Threads	The peak number of inactive threads of this DB2ENTRY that were protected. Source field: D2R-PTHREAD-HWM
Current number of DB2Entry Tasks	The current number of CICS tasks using this DB2ENTRY. Source field: D2R-TASK-CURRENT
Peak number of DB2Entry Tasks	The peak number of CICS tasks using this DB2ENTRY. Source field: D2R-TASK-HWM
Current Total number of DB2Entry Tasks	The current total number of tasks that have used this DB2ENTRY. Source field: D2R-TASK-TOTAL + D2R-TASK-CURRENT
Current number of Tasks on DB2Entry Readyq	The current number of CICS tasks waiting for a thread to become available for this DB2ENTRY. Source field: D2R-READYQ-CURRENT
Peak number of Tasks on DB2Entry Readyq	The peak number of CICS tasks that waited for a thread to become available for this DB2ENTRY. Source field: D2R-READYQ-HWM

DFHRPL and LIBRARY Analysis report

The DFHRPL and LIBRARY Analysis report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM**, **EXEC CICS COLLECT STATISTICS PROGRAM** and **EXEC CICS EXTRACT LIBRARY** commands. The statistics data was mapped by the DFHLDRDS and **DFHLDBDS DSECT**.

Table 235. Fields in the DFHRPL and LIBRARY Analysis report

Field Heading	Description
DFHRPL Offset	The offset into the DFHRPL DD program library concatenation.
DFHRPL Data set name	The name of the DFHRPL data set
Programs	The current number of programs, maps, and partitionsets defined to CICS and located in this concatenation of the static DFHRPL or dynamic program LIBRARY.
Times Used	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 static DFHRPL or dynamic program LIBRARY . Source field: LDRTU
Fetches	The number of times programs were fetched from this concatenation of the static DFHRPL or dynamic program LIBRARY. Source field: LDRFC
Average Fetch Time	The average fetch time for programs fetched from this concatenation of the static DFHRPL or dynamic program LIBRARY. Source field: (LDRFT / LDRFC)

Table 235. Fields in the DFHRPL and LIBRARY Analysis report (continued)

Field Heading	Description
Newcopies	The number of times programs were newcopied which have been fetched from this concatenation of the static DFHRPL or dynamic program LIBRARY. Source field: LDRTN
Removes	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 static DFHRPL or dynamic program LIBRARY. Source field: LDRRPC

Dispatcher reports

There are four dispatcher reports, the Dispatcher report, the Dispatcher MVS TCBs report, the Dispatcher TCB Modes report, and the Dispatcher TCB Pools report.

Dispatcher report

The Dispatcher report is produced using a combination of the **EXEC CICS INQUIRE SYSTEM** and **EXEC CICS EXTRACT STATISTICS DISPATCHER** commands. The statistics data is mapped by the **DFHDSGDS DSECT**.

Table 236. Fields in the Dispatcher Report

Field Heading	Description
Current ICV time	The current interval control value, expressed in milliseconds. Source field: DSGICVT
Current ICVR time	The current task runaway time interval, expressed in milliseconds. Source field: DSGICVRT
Current ICVTSD time	The current terminal scan delay value, expressed in milliseconds. With SNA and IPIC networks, the default ICVTSD value of 0 is appropriate. Source field: DSGICVSD
Current PRTYAGING time	The current task priority aging factor. Source field: DSGPRIAG
MRO (QR) Batching (MROBTCH) value	The number of events that must ocur before CICS is posted for dispatch due to the batching mechanism, as specified in the MROBTCH value in the SIT. Source field: DSGMBTCH
Concurrent Subtasking (SUBTSKS) value	The number of task control blocks (TCBs) that CICS can use for running tasks in concurrent mode, as specified in the SUBTSKS SIT parameter. Source field: DSGSTSKS
Current number of CICS Dispatcher tasks	The current number of tasks in the system. This figure includes all system tasks and all user tasks. Source field: DSGCNT
Peak number of CICS Dispatcher tasks	The peak number of tasks concurrently in the system. Source field: DSGPNT

Table 236. Fields in the Dispatcher Report (continued)

Field Heading	Description
Current number of TCBs attached	The current number of TCBs attached for this CICS address space. Source field: DSGTCBCA
Current number of TCBs in use	The number of CICS TCBs in use. Source field: DSGTCBCU
Last Excess TCB Scan	The date and time of the last CICS dispatcher excess MVS TCB scan. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that an excess TCB scan has not happened yet. Source field: DSGLXSCN Reset characteristic: not reset
Number of Excess TCB Scans	The number of excess TCB scans performed by the CICS dispatcher. Source field: DSGXSCNS
Last Excess TCB scan — No TCB Detached	The date and time of the last CICS dispatcher excess MVS TCB scan that did not detach any TCBs. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that an excess TCB scan has not happened yet. Source field: DSGLXSND Reset characteristic: not reset
Excess TCB scans — No TCB Detached	The number of excess TCB scans performed by the CICS dispatcher during which no CICS TCBs were detached. Source field: DSGXSCNN
Number of Excess TCBs Detached	The number of CICS TCBs that were detached by the CICS dispatcher during excess TCB scans. Source field: DSGXTCBD
Average Excess TCBs Detached per Scan	The average number of CICS TCBs that were detached by the CICS dispatcher during each excess TCB scan. Source field: DSGXTCBD / DSGXSCNS
Number of CICS TCB MODEs	The number of CICS TCB modes for this CICS address space. Source field: DSGASIZE
Number of CICS TCB POOLs	The number of CICS TCB pools for this CICS address space. Source field: DSGPSIZE

Dispatcher MVS TCBs report

The Dispatcher MVS TCBs report is produced using the **EXEC CICS EXTRACT STATISTICS MVSTCB**,**EXEC CICS EXTRACT STATISTICS DISPATCHER**, and **EXEC CICS INQUIRE MVSTCB** commands. The statistics data is mapped by the **DFHDSGDS**, **DFHDSTDS**, and **DFHDSRDS DSECT**.

Table 237. Fields in the Dispatcher MVS TCBs report

Field Heading	**Description**
Dispatcher Start Time and Date	The local time and date at which the CICS dispatcher started. Source field: DSGLSTRT
Address Space Accumulated CPU Time	The accumulated CPU time since reset for this CICS address space. **Note:** This field is not reset at CICS statistics intervals. Source field: MVS field ASCBEJST
Address Space Accumulated SRB Time	The accumulated SRB time since reset for this CICS address space. **Note:** This field is not reset at CICS statistics intervals. Source field: MVS field ASCBSRBT
Address Space CPU Time (Since Reset)	The accumulated CPU time for this CICS address space. Source field: DSGEJST
Address Space SRB Time (Since Reset)	The accumulated SRB time for this CICS address space. Source field: DSGSRBT
Current number of CICS TCBs	The current number of CICS TCBs in the address space. Source field: DSTDS_CICSTCB_COUNT
Current CICS TCB CPU time	The total CPU time so far for the currently attached CICS TCBs. Source field: DSTDS_CICSTCB_CPUTIME
Current CICS TCB Private Stg below 16MB	The total private storage below 16 MB allocated to CICS TCBs. Source field: DSTDS_CICSTCB_STG_BELOW
Current CICS TCB Private Stg below 16MB in use	The total private storage below 16 MB in use by CICS TCBs. [1] Source field: DSTDS_CICSTCB_STG_BELOW_INUSE
Current CICS TCB Private Stg above 16MB	The total private storage above 16 MB allocated to CICS TCBs. Source field: DSTDS_CICSTCB_STG_ABOVE
Current CICS TCB Private Stg above 16MB in use	The total private storage above 16 MB in use by CICS TCBs. [1] Source field: DSTDS_CICSTCB_STG_ABOVE_INUSE
Current number of non-CICS TCBs	The current number of non-CICS TCBs in the address space. Source field: DSTDS_NONCICSTCB_COUNT
Current non-CICS TCB CPU time	The total CPU time so far for the currently attached non-CICS TCBs. Source field: DSTDS_NONCICSTCB_CPUTIME
Current non-CICS TCB Private Stg below 16MB	The total private storage below 16 MB allocated to non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_BELOW
Current non-CICS TCB Private Stg below 16MB in use	The total private storage below 16 MB in use by non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_BELOW_INUSE
Current non-CICS TCB Private Stg above 16MB	The total private storage above 16 MB allocated to non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_ABOVE
Current non-CICS TCB Private Stg above 16MB in use	The total private storage above 16 MB in use by non-CICS TCBs. Source field: DSTDS_NONCICSTCB_STG_ABOVE_INUSE

Table 237. Fields in the Dispatcher MVS TCBs report (continued)

Field Heading	Description
TCB Address	The address of the MVS TCB. Source field: DSRDS_TCB_ADDRESS
TCB Name	The name of the MVS TCB (if known to CICS). Source field: DSRDS_TCB_NAME
CICS TCB	The type of TCB, CICS or non-CICS. Source field: DSRDS_TCB_TYPE
Current TCB CPU Time	The total CPU time so far for this TCB. Source field: DSRDS_TCB_CPUTIME
Current TCB Private Stg Below 16MB Allocated	The total private storage below 16 MB allocated to this TCB. Source field: DSRDS_TCB_STG_BELOW
Current TCB Private Stg Below 16MB In Use	The total private storage below 16 MB in use by this TCB. Source field: DSRDS_TCB_STG_BELOW_INUSE
Current TCB Private Stg Above 16MB Allocated	The total private storage above 16 MB allocated to this TCB. Source field: DSRDS_TCB_STG_ABOVE
Current TCB Private Stg Above 16MB In Use	The total private storage above 16 MB in use by this TCB. Source field: DSRDS_TCB_STG_ABOVE_INUSE
Task Number	The CICS task number currently associated with this TCB. None means there are no CICS transactions currently assigned to this TCB. Source field: DSRDS_TCB_CICS_TASK
Tran ID	Transaction ID of the task currently associated with this TCB, if any. Source field: **`EXEC CICS INQUIRE TASK() TRANSACTION()`**
Task Status	Status of the task currently associated with this TCB, if any. Source field: **`EXEC CICS INQUIRE TASK() RUNSTATUS()`**
Mother TCB	Address of mother TCB. Source field: DSRDS_TCB_MOTHER
Sister TCB	Address of sister TCB. Source field: DSRDS_TCB_SISTER
Daughter TCB	Address of daughter TCB. Source field: DSRDS_TCB_DAUGHTER

Note:

1. The statistics for storage in use show the amount of storage that tasks obtain by using GETMAIN requests. This might be less than the amount of storage allocated to the TCBs, because storage is always allocated to TCBs in page multiples (4096 bytes).

Dispatcher TCB Modes report

The Dispatcher TCB Modes report is produced using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** command. The statistics data is mapped by the **DFHDSGDS DSECT**.

In the Dispatcher TCB Modes report, some fields (for example, TCB Allocates) apply to open TCB modes only. The validity of these fields for each mode can be determined only after a TCB has been attached in that mode. Until the first TCB has been attached in that mode, the fields are marked “N/A”. After the first TCB has been attached in that mode, if it is not an open TCB mode, the field continues to be marked “N/A”. If it is an open TCB mode, the field is given a value.

Table 238. Fields in the Dispatcher TCB Modes report

Field Heading	Description
Dispatcher Start Time and Date	The local time and date at which the CICS dispatcher started. Source field: DSGLSTRT
Address Space Accumulated CPU Time	The accumulated CPU time since reset for this CICS address space. This field is not reset at CICS statistics intervals. Source field: MVS field ASCBEJST
Address Space Accumulated SRB Time	The accumulated SRB time since reset for this CICS address space. This field is not reset at CICS statistics intervals. Source field: MVS field ASCBSRBT
Address Space CPU Time (Since Reset)	The accumulated CPU time for this CICS address space. Source field: DSGEJST
Address Space SRB Time (Since Reset)	The accumulated SRB time for this CICS address space. Source field: DSGSRBT
TCB Mode	The name of the TCB mode to which the statistics refer. The names of the TCB modes are QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, and T8. Source field: DSGTCBNM
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
Op. System Waits	The number of MVS waits that occurred on this TCB. Source field: DSGSYSW
Op. System Wait Time	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
Total TCB Dispatch Time	The accumulated real time that this TCB has been dispatched by MVS; that is, the total time used between the end of an MVS wait issued by the dispatcher and the start of the subsequent wait issued by the dispatcher. Source field: DSGTDT

Table 238. Fields in the Dispatcher TCB Modes report (continued)

Field Heading	Description
Total TCB CPU Time	The accumulated CPU time taken for this TCB; that is, the total time that this TCB has been running. Source field: DSGACT
DS TCB CPU Time	The accumulated CPU time taken for this DS task; that is, the processor time used by this TCB while running the default dispatcher task (DSTCB). Source field: DSGTCT
TCB CPU/Disp Ratio	The ratio (expressed as a percentage) of the accumulated CPU time to accumulated dispatch time for this TCB. This ratio is calculated only for the QR TCB. Source field: ((DSGACT / DSGTDT) * 100)
TCBs attached — Current	The total number of TCBs currently attached. Source field: DSGTCBCA for each TCB mode
Total TCB CPU Time	The total accumulated CPU time taken for the active TCBs. Source field: DSGACT for each TCB mode
DS TCB CPU Time	The total accumulated CPU time taken for the DS task on each active dispatcher TCB. Source field: DSGTCT for each TCB mode
TCB Mode	The name of the TCB mode to which the statistics refer. The names of the TCB modes are QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, and T8. Source field: DSGTCBNM
Open	Indicates whether this TCB mode is an open TCB mode, not an open TCB mode, or unknown. Unknown means that this TCB mode has not been activated; the first request for a TCB in a particular mode causes the mode to be activated. Source field: DSGTCBMD
TCB Pool	The name of the TCB pool in which this TCB mode is defined: OPEN, SSL, THRD, XP, or N/A. Source field: DSGTCBMP
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
TCBs In Use - Current	The current number of TCBs in use in this mode. Source field: DSGTCBCU
TCBs In Use - Peak	The peak number of TCBs in use in this mode. Source field: DSGTCBPU
TCB Allocates	The number of times a TCB from this TCB mode was allocated to a task; that is, CICS assigned the TCB for the use of a particular task. TCB allocates apply only to open TCB modes. "N/A" means either that this TCB mode is not open or that no TCBs have yet been created in this mode. Source field: DSGTCBAL

Table 238. Fields in the Dispatcher TCB Modes report (continued)

Field Heading	Description
Dispatchable Queue - Current	The current number of dispatchable tasks queued for the TCB. Source field: DSGTMCDQ Reset characteristic: not reset
Dispatchable Queue - Peak	The peak number of dispatchable tasks that have been queued for the TCB. Source field: DSGTMPDQ Reset characteristic: reset to current
Dispatchable Queue - Average	The average number of dispatchable tasks that have been queued for the TCB. Source field: DSGTMADQ Reset characteristic: reset to current
TCBs Attached - Current	The total number of TCBs currently attached for all modes. Source field: DSGTCBCA for each TCB mode
TCBs In Use - Current	The total number of TCBs currently in use for all modes. Source field: DSGTCBCU for each TCB mode
TCB Allocates	The total number of times a TCB from this TCB mode was allocated to a task. Source field: DSGTCBAL for each TCB mode
TCB Mode	The name of the TCB mode to which the statistics refer. The names of the TCB modes are QR, RO, CO, SZ, RP, FO, SL, SO, SP, EP, TP, D2, S8, L8, L9, X8, X9, and T8. Source field: DSGTCBNM
Open	Indicates whether this TCB mode is an open TCB mode, not an open TCB mode, or unknown. Unknown means that this TCB mode has not been activated; the first request for a TCB in a particular mode will cause the mode to be activated. Source field: DSGTCBMD
TCB Pool	The name of the TCB pool in which this TCB mode is defined: OPEN, SSL, THRD, XP, or N/A. Source field: DSGTCBMP
TCB Attaches	The total number of MVS TCB attaches in this mode. Source field: DSGNTCBA
Attach Failures	The number of MVS TCB attach failures that have occurred in this mode. Source field: DSGTCBAF
TCBs Detached - Unclean	The number of MVS TCBs that have been, or are in the process of being, detached for this CICS dispatcher mode because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU
TCBs Detached - Stolen	The number of MVS TCBs that have been, or are in the process of being, stolen from this CICS dispatcher mode because it is required by another TCB mode. Source field: DSGTCBDS

Table 238. Fields in the Dispatcher TCB Modes report (continued)

Field Heading	Description
TCBs Detached - Excess	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher mode because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX
TCBs Detached - Other	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode for other reasons, for example, because too many TCBs are attached in relation to the number of TCBs in use. Source field: DSGTCBDO
TCB Steals	The number of MVS TCBs that have been stolen from other TCB modes. Source field: DSGTCBST
TCB Mismatches	The number of TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM
TCB Attaches	The total number of TCB attaches for all modes. Source field: DSGNTCBA for each TCB mode
Attach Failures	The total number of MVS TCB attach failures that have occurred in this mode. Source field: DSGTCBAF
TCBs Detached - Unclean	The total number of MVS TCBs that have been, or are in the process of being, detached because the CICS transaction associated with the TCB has abended, for all modes. Source field: DSGTCBDU for each TCB mode
TCBs Detached - Stolen	The total number of MVS TCBs that have been, or are in the process of being, stolen because they are required by another TCB mode, for all modes. Source field: DSGTCBDS for each TCB mode
TCBs Detached - Excess	The total number of MVS TCBs that have been, or are in the process of being, detached because of the CICS dispatcher excess TCB scans, for all modes. Source field: DSGTCBDX for each TCB mode
TCBs Detached - Other	The total number of MVS TCBs that have been, or are in the process of being, detached for other reasons, for all modes. Source field: DSGTCBDO for each TCB mode
TCB Steals	The total number of MVS TCBs that have been stolen from other TCB modes, for all modes. Source field: DSGTCBST for each TCB mode
TCB Mismatches	The total number of TCB mismatches that have occurred for all TCB modes. Source field: DSGTCBMM for each TCB mode

Dispatcher TCB Pools report

The Dispatcher TCB Pools report is produced for each TCB pool. The example shows the OPEN TCB pool. This report is produced using the **EXEC CICS EXTRACT STATISTICS DISPATCHER** command. The statistics data is mapped by the **DFHDSGDS DSECT**.

Table 239. Fields in the Dispatcher TCB Pools report

Field Heading	Description
TCB Pool	The name of the CICS TCB pool, either OPEN, SSL, THRD, or XP. Source field: DSGTCBPN
Current TCBs attached in this TCB Pool	The current number of TCBs attached in this TCB pool. Source field: DSGCNUAT
Peak TCBs attached in this TCB Pool	The peak number of TCBs attached in this TCB pool. Source field: DSGPNUAT
Current TCBs in use in this TCB Pool	The current number of TCBs in use in this TCB pool. Source field: DSGCNUUS
Peak TCBs in use in this TCB Pool	The peak number of TCBs in use in this TCB pool. Source field: DSGPNUUS
Max TCB Pool Limit	The value for the maximum number of TCBs allowed in this pool: • The **MAXOPENTCBS** system initialization parameter, if specified, sets the value for the open TCB pool. If the MAXOPENTCBS system initialization is not specified, CICS sets the limit for the L8 and L9 mode open TCB pool automatically based on the maximum number of tasks specified for the CICS region (the MXT value), using the following formula: `(2 * MXT Value) + 32`. • The **MAXSSLTCBS** system initialization parameter specifies the value for the SSL TCB pool. • The **MAXTHRDTCBS** system initialization parameter specifies the value for the JVM server THRD TCB pool. The number of threads reserved for each JVM server is the THREADLIMIT value on the JVMSERVER resource, plus 1, up to a limit of 2000. • The **MAXXPTCBS** system initialization parameter, if specified, sets the value for the XP TCB pool. If the **MAXXPTCBS** system initialization is not specified, CICS sets the limit for the X8 and X9 mode XP TCB pool automatically to a value equal to the maximum number of tasks specified for the CICS region (the MXT value) Source field: DSGMXTCB
Times at Max TCB Pool Limit	The number of times the system reached the limit for the number of TCBs allowed in this pool: • OPEN TCB pool • SSL TCB pool • THRD TCB pool • XP TCB pool Source field: DSGNTCBL
Time Pool Limit last reached	The time at which the pool reached the maximum TCB limit. Source field: DSGLTCBL Reset characteristic: reset to zero
Requests Delayed by Max TCB Pool Limit	The total number of TCB attaches delayed because the system reached the limit for the number of TCBs allowed in this pool. Source field: DSGTOTNW

Table 239. Fields in the Dispatcher TCB Pools report (continued)

Field Heading	Description
Total Max TCB Pool Limit delay time	The total time that TCB requests were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: DSGTOTWL
Average Max TCB Pool Limit delay time	The average time that a TCB request was delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: (DSGTOTWL and DSGTOTNW)
Current Requests Delayed by Max TCB Pool Limit	The number of TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: DSGCURNW
Peak Requests Delayed by Max TCB Pool Limit	The peak number of TCB requests that were delayed because the system had reached the limit for the number of TCBs allowed in this pool. Source field: DSGPEANW
Total Delay Time for current delayed	The total delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: DSGCURWT
Average Delay time for current delayed	The average delay time for the TCB requests that are currently delayed because the system has reached the limit for the number of TCBs allowed in this pool. Source field: (DSGCURWT and DSGCURNW)
Total number of TCB Mismatch Waits	The total number of TCB mismatch waits; that is, TCB requests that waited because no available TCB matched the request, but at least one non-matching TCB was free. Source field: DSGMMWTS
Total TCB Mismatch wait time	The total time spent in TCB mismatch waits by TCB requests using this pool. Source field: DSGMMWTM
Average TCB Mismatch wait time	The average time spent in a TCB mismatch wait by TCB requests using this pool. Source field: (DSGMMWTM and DSGMMWTS)
Current TCB Mismatch Waits	The current number of TCB mismatch waits by TCB requests using this pool. Source field: DSGCMMWS
Peak TCB Mismatch Waits	The peak number of TCB mismatch waits by TCB requests using this pool. Source field: DSGPMMWS
Total Wait time for current Mismatch Waits	The total wait time for current TCB mismatch waits by TCB requests using this pool. Source field: DSGCMMWT
Average Wait time for current Mismatch Waits	The average wait time for current TCB mismatch waits by TCB requests using this pool. Source field: (DSGCMMWT and DSGCMMWS)
Requests Delayed by MVS storage constraint	The total number of TCB requests that waited because no TCB was available, and none was created because of MVS storage constraints. Source field: DSGTOTMW

Table 239. Fields in the Dispatcher TCB Pools report (continued)

Field Heading	Description
Total MVS storage constraint delay time	The total time spent in waits caused by MVS storage constraints for TCB requests using this pool. Source field: DSGTOTMT
Average MVS storage constraint delay time	The average time spent in waits caused by MVS storage constraints for TCB requests using this pool. Source field: (DSGTOTMT and DSGTOTMW)
TCB Mode	The TCB modes currently active in this TCB Pool. The report states if no TCB modes are active. Source field: DSGTCBNM
TCBs Attached - Current	The current number of TCBs attached in this mode. Source field: DSGTCBCA
TCBs Attached - Peak	The peak number of TCBs attached in this mode. Source field: DSGTCBPA
TCBs In Use - Current	The current number of TCBs in use in this mode. Source field: DSGTCBCU
TCBs In Use - Peak	The peak number of TCBs in use in this mode. Source field: DSGTCBPU
TCB Attaches	The total number of MVS TCB attaches for this mode. Source field: DSGNTCBA
TCBs Detached - Unclean	The number of MVS TCBs that have been, or are in the process of being, detached for this CICS dispatcher mode because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU
TCBs Detached - Stolen	The number of MVS TCBs that have been, or are in the process of being, stolen from this CICS dispatcher mode because it is required by another TCB mode. Source field: DSGTCBDS
TCBs Detached - Excess	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher mode because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX
TCBs Detached - Other	The number of MVS TCBs that have been, or are in the process of being, detached from this CICS dispatcher TCB mode for other reasons; for example, because the TCB pool limit has been lowered, or because there are too many TCBs attached in relation to the number of TCBs in use. Source field: DSGTCBDO
TCB Steals	The number of MVS TCBs that have been stolen from other TCB modes. Source field: DSGTCBST
TCB Mismatches	The number of MVS TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM

Table 239. Fields in the Dispatcher TCB Pools report (continued)

Field Heading	Description
TCBs Attached - Current	The total number of TCBs currently attached in this TCB pool for all modes. Source field: DSGTCBCA for each TCB mode
TCBs In Use - Current	The total number of TCBs currently in use in this TCB pool for all modes. Source field: DSGTCBCU for each TCB mode
TCB Attaches	The total number of MVS TCB attaches in this TCB pool for all modes. Source field: DSGNTCBA for each TCB mode
TCBs Detached - Unclean	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, detached because the CICS transaction associated with the TCB has abended. Source field: DSGTCBDU for each TCB mode
TCBs Detached - Stolen	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, stolen from a CICS dispatcher mode because they are required by another TCB mode. Source field: DSGTCBDS for each TCB mode
TCBS Detached - Excess	The total number of MVS TCBs in this TCB pool that have been or are in the process of being, detached because of the CICS dispatcher excess TCB scans. Source field: DSGTCBDX for each TCB mode
TCBs Detached - Other	The total number of MVS TCBs in this TCB pool that have been, or are in the process of being, detached for other reasons. Source field: DSGTCBDO for each TCB mode
TCB Steals	The total number of MVS TCBs in this TCB pool that have been stolen from other TCB modes. Source field: DSGTCBST for each TCB mode
TCB Mismatches	The number of MVS TCB mismatches that have occurred for this TCB mode. Source field: DSGTCBMM for each TCB mode

Document Templates report

The Document Templates report is produced using the **EXEC CICS EXTRACT STATISTICS DOCTEMPLATE** command and the **EXEC CICS INQUIRE DOCTEMPLATE** command. The statistics data is mapped by the DFHDHDDS DSECT.

Table 240. Fields in the Document Templates report

Field Heading	Description
DOCTEMPLATE Name	The name of the DOCTEMPLATE resource definition. Source field: **EXEC CICS INQUIRE DOCTEMPLATE**
Template Name	The name by which the template is known to application programs (the TEMPLATENAME attribute in the DOCTEMPLATE resource definition). Source field: DHD-TEMPLATE-NAME

Table 240. Fields in the Document Templates report (continued)

Field Heading	Description
Append crlf	Whether CICS appends carriage-return line-feed to each logical record of the template. Source field: DHD-APPEND-CRLF
Template contents	The format of the contents of the template, either binary or EBCDIC. Source field: DHD-TEMPLATE-CONTENTS
Template cache size	The amount of storage required for a cached copy of the document template. • Before the first use of the template, this field is zero. • This field is always zero for templates in a CICS program, which are never cached, and for templates in an exit program if they are not specified for caching. Source field: DHD-TEMPLATE-CACHE-SIZE
Template type	The type for the source of the document template, which can be an exit program, a CICS file name for a data set, a zFS file, a member of a PDS, a program, a transient data queue, or a temporary storage queue. Source field: DHD-TEMPLATE-TYPE
[Template type] name	The name for the source of the document template, such as a program name or z/OS UNIX file name. Source field: one of DHD-TEMPLATE-EXIT-PROGRAM, DHD-TEMPLATE-FILE-NAME, DHD-TEMPLATE-PROGRAM-NAME, DHD-TEMPLATE-PDS-MEMBER, DHD-TEMPLATE-TDQUEUE, DHD-TEMPLATE-TSQUEUE, DHD-TEMPLATE-HFSFILE
Data set name	Only for document templates of type "File". The name of the data set containing the document template. Source field: **`EXEC CICS INQUIRE FILE() DSNAME()`**
PDS Data set name	Only for document templates of type "PDS". The name of the partitioned data set containing the document template. Source field: **`EXEC CICS INQUIRE DOCTEMPLATE() DSNAME()`**
Use count	The total number of times the document template was referenced for any reason. Source field: DHD-TEMPLATE-USE-COUNT
Newcopy count	The number of times the SET DOCTEMPLATE NEWCOPY command was issued for this document template. Source field: DHD-TEMPLATE-NEWCOPIES
Read count	The number of times the document template was read from the source. This happens on the first use (including the first reference after deletion from the cache), or by a SET DOCTEMPLATE NEWCOPY command. Source field: DHD-TEMPLATE-READ-COUNT
Cache copy used	The number of times an application used the cached copy of the document template. Source field: DHD-TEMPLATE-CACHE-USED
Cache copy deleted	The number of times the cached copy of the document template was deleted because of a short-on-storage condition. Source field: DHD-TEMPLATE-CACHE-DELETED

Enqueue reports

There are two enqueue reports, the Enqueue Manager report, and the Enqueue Models report.

Enqueue Manager report

The Enqueue Manager report is produced using the **EXEC CICS EXTRACT STATISTICS ENQUEUE** command. The statistics data is mapped by the **DFHNQGDS DSECT**.

Table 241. Fields in the Enqueue Manager report

Field Heading	Description
ENQueue poolname	The enqueue pool name. Source field: NQGPOOL
ENQs issued	The number of enqueues issued. Source field: NQGTNQSI
ENQs waited	The number of enqueues that waited. Source field: NQGTNQSW
ENQueue waiting time	The total enqueue waiting time for the enqueues that waited. Source field: NQGTNQWT
Average Enqueue wait time	The average enqueue wait time. Source field: NQGTNQWT / NQGTNQSW
Current ENQs waiting	The current number of ENQs waiting. Source field: NQGCNQSW
Current ENQueue waiting time	The total enqueue waiting time for the ENQs currently waiting. Source field: NQGCNQWT
Sysplex ENQs waited	The number of sysplex enqueues that waited. Source field: NQGGNQSW
Sysplex ENQueue waiting time	The total sysplex enqueue waiting time for the sysplex enqueues that waited. Source field: NQGGNQWT
Average Sysplex Enqueue wait time	The average sysplex enqueue wait time. Source field: NQGGNQWT / NQGGNQSW
Current Sysplex ENQs waiting	The current number of sysplex enqueues waiting. Source field: NQGSNQSW
Current Sysplex ENQueue waiting time	The total enqueue waiting time for the sysplex ENQs currently waiting. Source field: NQGSNQWT
Total ENQs retained	The total number of enqueues retained. Source field: NQGTNQSR
Enqueue retention time	The total enqueue retention time. Source field: NQGTNQRT

Table 241. Fields in the Enqueue Manager report (continued)

Field Heading	Description
Average Enqueue retention time	The average enqueue retention time. Source field: NQGTNQRT / NQGTNQSR
Current ENQs retained	The current number of enqueues retained. Source field: NQGCNQSR
Current Enqueue retention time	The total enqueue retention time for enqueues currently retained. Source field: NQGCNQRT
Current Average Enqueue retention time	The current average enqueue retention time. Source field: NQGCNQRT / NQGCNQSR
Enqueues Rejected - Enqbusy	The number of enqueues rejected immediately - ENQBUSY. Source field: NQGTIRJB
Enqueues Rejected - Retained	The number of immediately rejected retained enqueues. Source field: NQGTIRJR
Waiting Enqueues - Rejected Retained	The number of retained enqueues awaiting rejection. Source field: NQGTWRJR
Waiting Enqueues Purged - Operator	The number of enqueues awaiting rejection because of operator intervention. Source field: NQGTWPOP
Waiting Enqueues Purged - Timeout	The number of enqueues awaiting rejection because of timeout. Source field: NQGTWPTO

Enqueue Models report

The Enqueue Models report is produced using the **EXEC CICS INQUIRE ENQMODEL** command.

Table 242. Fields in the Enqueue Models report

Field Heading	Description
ENQModel Name	The name (identifier) of the enqueue model. Source field: **EXEC CICS INQUIRE ENQMODEL()**
ENQModel Enqname	The resource name or generic name for this enqueue model. Source field: **EXEC CICS INQUIRE ENQMODEL() ENQNAME()**
ENQModel Enqscope	Indicates whether the enqueue is local or sysplex-wide. Source field: **EXEC CICS INQUIRE ENQMODEL() ENQSCOPE()**
ENQModel Status	The current status of this enqueue. Source field: **EXEC CICS INQUIRE ENQMODEL() STATUS(cvda)**

Event processing reports

There are four event processing reports, the CAPTURESPEC report, the EPADAPTER report, the EVENTBINDING report and the EVENTPROCESS report.

CAPTURESPEC report

The CAPTURESPEC report shows information and statistics about the capture specifications for each event. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING**, **EXEC CICS INQUIRE CAPTURESPEC**, **EXEC CICS EXTRACT STATISTICS EVENTBINDING**, and **CAPTURESPEC** commands.

The statistics data is mapped by the DFHECCDS DSECT.

Table 243. Fields in the CAPTURESPEC report

Field Heading	Description
EVENTBINDING Name	The name of the associated event binding. Source field: **EXEC CICS INQUIRE EVENTBINDING**
EPADAPTER Name	The 32-character name of an event binding. Source field: **EXEC CICS INQUIRE EVENTBINDING**
Enable Status	The current enable status of the event binding. Source field: **EXEC CICS INQUIRE EVENTBINDING ENABLESTATUS()**
CAPTURESPEC name	The name of the capture specification. Source field: **EXEC CICS INQUIRE CAPTURESPEC**
Capture point	The capture point associated with the capture specification. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CAPTURETYPE** and **EXEC CICS INQUIRE CAPTURESPEC CAPTUREPOINT**
Current Program	The value of the current program application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRPGM**
Current Program Op	The value of the operator for the current program application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRPGMOP**
Current Transaction	The value of the current transaction application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRTRANID**
Current Transaction Op	The value of the operator for the current transaction application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRTRANIDOP**
Current Userid	The value of the current user ID application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRUSERID**
Current Userid Op	The value of the operator for the current user ID application context predicate. Source fields: **EXEC CICS INQUIRE CAPTURESPEC CURRUSERIDOP**
Event name	The associated business event name. Source field: **EXEC CICS INQUIRE CAPTURESPEC EVENTNAME**
Events Captured	The total number of events captured. Source field: ECC-EVENTS-CAPTURED
Capture Failures	The number of capture failures, recorded by capture specification. When displayed, this statistic is totaled by event binding. Source field: ECC-CAPTURE-FAILURES

Related reference:

"EPADAPTER report"
The EPADAPTER report shows information and statistics about each EP adapter. This report is produced using a combination of **EXEC CICS INQUIRE EPADAPTER** and **EXEC CICS EXTRACT STATISTICS EPADAPTER** commands.

"EVENTBINDING report" on page 831
The EVENTBINDING report shows information and statistics about each event binding and the event binding status. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING** and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

"EVENTPROCESS report" on page 831
The EVENTPROCESS report shows information and statistics about event processing, queue status, tasks, and number of events captured. This report is produced using a combination of **EXEC CICS INQUIRE EVENTPROCESS**, **EXEC CICS EXTRACT STATISTICS EVENTPROCESS**, and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

EPADAPTER report

The EPADAPTER report shows information and statistics about each EP adapter. This report is produced using a combination of **EXEC CICS INQUIRE EPADAPTER** and **EXEC CICS EXTRACT STATISTICS EPADAPTER** commands.

The statistics data is mapped by the DFHEPRDS DSECT.

Table 244. Fields in the EPADAPTER report

Field Heading	Description
EPADAPTER name	The name of the EP adapter. Source field: EPR-ADAPTER-NAME
Enable status	The current enable status of the EP adapter. Source field: **EXEC CICS INQUIRE EPADAPTER ENABLESTATUS()**
EPADAPTER Type	The adapter type. Source field: EPR-ADAPTER-TYPE
EPADAPTER Emission mode	The EP adapter emission mode. This identifies whether the EP adapter is for synchronous or asynchronous events. Source field: EPR-EMISSION-MODE
EPADAPTER Number of put events	The number of events passed to EP for emission by this adapter. Source field: EPR-PUT-EVENTS

Related reference:

"CAPTURESPEC report" on page 829
The CAPTURESPEC report shows information and statistics about the capture specifications for each event. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING**, **EXEC CICS INQUIRE CAPTURESPEC**, **EXEC CICS EXTRACT STATISTICS EVENTBINDING**, and **CAPTURESPEC** commands.

"EVENTBINDING report" on page 831
The EVENTBINDING report shows information and statistics about each event binding and the event binding status. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING** and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

"EVENTPROCESS report"
The EVENTPROCESS report shows information and statistics about event processing, queue status, tasks, and number of events captured. This report is produced using a combination of **EXEC CICS INQUIRE EVENTPROCESS**, **EXEC CICS EXTRACT STATISTICS EVENTPROCESS**, and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

EVENTBINDING report

The EVENTBINDING report shows information and statistics about each event binding and the event binding status. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING** and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

The statistics data is mapped by the DFHECGDS DSECT.

Table 245. Fields in the EVENTBINDING report

Field Heading	Description
EVENTBINDING Name	The 32-character name of an event binding. Source field: **EXEC CICS INQUIRE EVENTBINDING**
EVENTBINDING EPADAPTER Name	The 32-character name of an EP adapter. Source field: **EXEC CICS INQUIRE EVENTBINDING**
Enable Status	The current enable status of the event binding. Source field: **EXEC CICS INQUIRE EVENTBINDING ENABLESTATUS()**

Related reference:

"CAPTURESPEC report" on page 829
The CAPTURESPEC report shows information and statistics about the capture specifications for each event. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING**, **EXEC CICS INQUIRE CAPTURESPEC**, **EXEC CICS EXTRACT STATISTICS EVENTBINDING**, and **CAPTURESPEC** commands.

"EPADAPTER report" on page 830
The EPADAPTER report shows information and statistics about each EP adapter. This report is produced using a combination of **EXEC CICS INQUIRE EPADAPTER** and **EXEC CICS EXTRACT STATISTICS EPADAPTER** commands.

"EVENTPROCESS report"
The EVENTPROCESS report shows information and statistics about event processing, queue status, tasks, and number of events captured. This report is produced using a combination of **EXEC CICS INQUIRE EVENTPROCESS**, **EXEC CICS EXTRACT STATISTICS EVENTPROCESS**, and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

EVENTPROCESS report

The EVENTPROCESS report shows information and statistics about event processing, queue status, tasks, and number of events captured. This report is produced using a combination of **EXEC CICS INQUIRE EVENTPROCESS**, **EXEC CICS EXTRACT STATISTICS EVENTPROCESS**, and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

The statistics data is mapped by the DFHEPGDS and DFHECGDS DSECTs.

Table 246. Fields in the EVENTPROCESS report

Field heading	Description
Event processing status	The current status of event processing. Source field: **EXEC CICS INQUIRE EVENTPROCESS**
Put events	The number of events passed to the EP component for emission. Source field: EPG-PUT-EVENTS
Commit forward events	The number of units of work that have been committed, and that included one or more asynchronous transactional events. Source field: EPG-COMMIT-FORWARD-EVENTS
Commit backward events	The number of units of work that have been backed out, and that included one or more asynchronous transactional events. Source field: EPG-COMMIT-BACKWARD-EVENTS
Current event capture queue	The current number of events on the event capture queue. Source field: EPG-CURRENT-EVC-QUEUE
Peak event capture queue	The peak number of events on the event capture queue. Source field: EPG-PEAK-EVC-QUEUE
Current transactional queue	The current number of events on the transactional queue. Source field: EPG-CURRENT-TRANS-QUEUE
Peak transactional queue	The peak number of events on the transactional queue. Source field: EPG-PEAK-TRANS-QUEUE
Async normal events	The number of asynchronous normal priority events. Source field: EPG-ASYNC-NORMAL-EVENTS
Async priority events	The number of asynchronous high priority events. Source field: EPG-ASYNC-PRIORITY-EVENTS
Transactional events	The number of transactional events. Source field: EPG-TRANS-EVENTS
Transactional events discarded	The number of transactional events discarded. Source field: EPG-TRANS-EVENTS-DISCARDED
Synchronous events	The number of synchronous emission events captured. Source field: EPG-SYNC-EVENTS
Synchronous events failed	The number of synchronous emission events that were not emitted. Source field: EPG-SYNC-EVENTS-FAILED
Dispatcher tasks attached	The number of dispatcher tasks attached. Source field: EPG-DISPATCHERS-ATTACHED
Current dispatcher tasks	The current number of dispatcher tasks. Source field: EPG-CURRENT-DISPATCHERS
Peak dispatcher tasks	The peak number of dispatcher tasks. Source field: EPG-PEAK-DISPATCHERS

Table 246. Fields in the EVENTPROCESS report (continued)

Field heading	Description
Events to WebSphere MQ EP adapter	The number of events dispatched to the WebSphere MQ EP adapter. Source field: EPG-WMQ-ADAPTER-EVENTS
Events to transaction EP adapter	The number of events dispatched to the Transaction EP adapter. Source field: EPG-TRANS-ADAPTER-EVENTS
Events to tsqueue EP adapter	The number of events dispatched to the TS queue EP adapter. Source field: EPG-TSQ-ADAPTER-EVENTS
Events to custom EP adapter	The number of events dispatched to the Custom EP adapter. Source field: EPG-CUSTOM-ADAPTER-EVENTS
Events to HTTP EP adapter	The number of events dispatched to the HTTP EP adapter. Source field: EPG-HTTP-ADAPTER-EVENTS
Events lost (dispatcher) - config	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem relating to a resource specified in the `eventDispatcherPolicy` section of the event binding. Source field: EPG-DISPATCH-FAILURE-CONFIG
Events lost (dispatcher) - other	The number of events that were captured but not dispatched to an EP adapter because the dispatcher encountered a problem in the CICS environment, for example, insufficient storage. Source field: EPG-DISPATCH-FAILURE-OTHER
Events lost (adapter) - config	The number of events that were captured but not emitted because the EP adapter encountered a problem relating to a resource specified in the `eventDispatcherAdapter` configuration section of the event binding. Source field: ECG-EVENTS-LOST-CONFIG
Events lost (adapter) - other	The number of events that were captured but not emitted because the EP adapter encountered a problem in the CICS environment, for example, insufficient storage. Source field: ECG-EVENTS-LOST-OTHER
Events lost - adapter unavailable	The number of events that were not emitted because the EP adapter is disabled or not installed. Source field: EPG-EVENTS-ADAPTER-UNAVAIL
Event filtering operations	The number of event filtering operations. Source field: ECG-EB-EVENT-FILTER-OPS
Events with disabled EVENTBINDING	The number of events that were not captured because of a disabled event binding. Source field: ECG-EB-EVENTS-DISABLED
Events captured	The total number of application and system events captured. Source field: ECG-EB-EVENTS-CAPTURED
System events captured	The number of system events captured. Source field: ECG-SYS-EVENTS-CAPTURED
Filter operations failed	The number of filtering operations that did not complete because CICS was unable to determine whether an event should have been captured. Source field: ECG-FILTER-OPS-FAILED

Table 246. Fields in the EVENTPROCESS report (continued)

Field heading	Description
Capture operations failed	The number of capture operations that did not complete because CICS determined that an event was required but failed to capture it. Source field: ECG-CAPTURE-OPS-FAILED

Related reference:

"CAPTURESPEC report" on page 829
The CAPTURESPEC report shows information and statistics about the capture specifications for each event. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING**, **EXEC CICS INQUIRE CAPTURESPEC**, **EXEC CICS EXTRACT STATISTICS EVENTBINDING**, and **CAPTURESPEC** commands.

"EPADAPTER report" on page 830
The EPADAPTER report shows information and statistics about each EP adapter. This report is produced using a combination of **EXEC CICS INQUIRE EPADAPTER** and **EXEC CICS EXTRACT STATISTICS EPADAPTER** commands.

"EVENTBINDING report" on page 831
The EVENTBINDING report shows information and statistics about each event binding and the event binding status. This report is produced using a combination of **EXEC CICS INQUIRE EVENTBINDING** and **EXEC CICS EXTRACT STATISTICS EVENTBINDING** commands.

Files report

The Files report is produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

Table 247. Fields in the Files report

Field Heading	Description
Filename	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(cvda)**
File Type	Indicates how the records are organized in the data set that corresponds to this file. Source field: **EXEC CICS INQUIRE FILE() TYPE(cvda)**
Remote Filename	The name by which the file is known in the remote system. Source field: **EXEC CICS INQUIRE FILE() REMOTENAME()**
Remote System	The name of the CICS region in which the file is defined. Source field: **EXEC CICS INQUIRE FILE() REMOTESYSTEM()**
LSRpool	The identity of the LSR pool defined for this file. "No" means that it is not defined in an LSR pool. Source field: **EXEC CICS INQUIRE FILE() LSRPOOLNUM()**
RLS	Indicates whether the file is to be opened in RLS mode. Source field: A17RLS

Table 247. Fields in the Files report (continued)

Field Heading	Description
Data Table Type	The type of data table: coupling facility, CICS-maintained, user-maintained, or remote. If this field is blank, it indicates that the file is not known to be defined as a data table. This can be the case if the file is not currently open. Source field: **EXEC CICS INQUIRE FILE() TABLE(cvda) REMOTETABLE(cvda)**
CFDT Poolname	The name of the coupling facility data table pool in which the coupling facility data table resides. Source field: **EXEC CICS INQUIRE FILE() CFDTPOOL()**
Table Name	The coupling facility data table name. Source field: **EXEC CICS INQUIRE FILE() TABLENAME()**
Recovery Status	Indicates the recovery status of the file. Source field: **EXEC CICS INQUIRE FILE() RECOVSTATUS(cvda)**
Strings	The number of VSAM strings that are defined for the file. Source field: A17STRNO
Buffers — Index	The number of index buffers that are defined for the file. Source field: A17DSINB
Buffers — Data	The number of data buffers that are defined for the file. Source field: A17DSDNB

File Requests report

The File Requests report is produced using a combination of the **EXEC CICS INQUIRE FILE** and **EXEC CICS EXTRACT STATISTICS FILE** commands. The statistics data is mapped by the **DFHA17DS DSECT**.

Table 248. Fields in the File Requests report

Field Heading	Description
Filename	The name of the file. Source field: **EXEC CICS INQUIRE FILE()**
Read Requests	The number of GET requests attempted against this file. Source field: A17DSRD
Get Update Requests	The number of GET UPDATE requests attempted against this file. Source field: A17DSGU
Browse Requests	The number of GETNEXT and GETPREV requests attempted against this file. Source field: A17DSBR
Browse Updates	The number of GETNEXT UPDATE and GETPREV UPDATE requests attempted against this file. Source field: A17DSBRU
Add Requests	The number of PUT requests attempted against this file. Source field: A17DSWRA

Table 248. Fields in the File Requests report (continued)

Field Heading	Description
Update Requests	The number of PUT UPDATE requests attempted against this file. Source field: A17DSWRU
Delete Requests	The number of DELETE requests attempted against this file. Source field: A17DSDEL
RLS Req. Timeouts	The number of RLS file requests that timed out. Source field: A17RLSWT
String Waits: Total	The total number of waits for strings against the file. Source field: A17DSTSW
String Waits: HWM	The peak number of waits for strings against the file. Source field: A17DSHSW

Global User Exits report

The Global User Exits report is produced using the **EXEC CICS INQUIRE EXITPROGRAM** command.

Table 249. Fields in the Global User Exits report

Field Heading	Description
Exit Name	The name of the global user exit point. Source field: **EXEC CICS INQUIRE EXITPROGRAM() EXIT()**
Program Name	The name of the exit program enabled at this global user exit point. Source field: **EXEC CICS INQUIRE EXITPROGRAM()**
Entry Name	The name of the entry point for this exit program at this global user exit point. Source field: **EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()**
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: **EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()**
Global Area Length	The length of the global work area for this exit program. Source field: **EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()**
Global Area Use Count	The number of exit programs that are associated with the global work area owned by this exit program. Source field: **EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()**
Number of Exits	The number of global user exit points at which this exit program is enabled. Source field: **EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()**
Program Status	Indicates whether this exit program is available for execution. Source field: **EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)**
Program Concurrency	Indicates the concurrency attribute of this exit program. Source field: **EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda)**

Table 249. Fields in the Global User Exits report (continued)

Field Heading	Description
Concurrency Status	Indicates the concurrency status of this exit program. It takes into account the fact that the PROGRAM definition may have been overriden by options on the ENABLE command. Source field: **EXEC CICS INQUIRE EXITPROGRAM() CONCURRENCY(cvda)**

IPCONN report

The IPCONN report shows information and statistics about IPCONN resource definitions, which define IP interconnectivity (IPIC) connections.

The IPCONN report is produced using a combination of the **EXEC CICS INQUIRE IPCONN** and **EXEC CICS EXTRACT STATISTICS IPCONN** commands. The statistics data is mapped by the DFHISRDS DSECT.

Table 250. Fields in the IPCONN report

Field Heading	Description
IPCONN Name	The name of the IPCONN definition; that is, the name by which CICS knows the remote system. Source field: ISR-IPCONN-NAME
IPCONN Applid	The application identifier (APPLID) of the remote system. If the remote system is a CICS region, its APPLID is specified on the APPLID parameter of its system initialization table. Source field: ISR-APPLID
IPCONN Status	The state of the connection between CICS and the remote system; for example, Acquired, Freeing, Obtaining, or Released. Source field: **EXEC CICS INQUIRE IPCONN() CONNSTATUS()**
IPCONN Port Number	The port number used for outbound requests on this IP connection; that is, the number of the port on which the remote system is listening. Source field: **EXEC CICS INQUIRE IPCONN() PORT()**
IPCONN Host	The host name of the remote system or its IPv4 or IPv6 address. Source field: **EXEC CICS INQUIRE IPCONN() HOST()**
IPCONN IP Resolved Address	The IPv4 or IPv6 resolved address of the host. Source field: **EXEC CICS INQUIRE IPCONN() IPRESOLVED()**
IPCONN IP Family	The address format of the address returned in IPCONN IP Resolved Address. Source field: **EXEC CICS INQUIRE IPCONN() IPFAMILY()**
SSL Authentication	Whether secure socket layer (SSL) authentication is supported: Yes No Source field: ISR-SSL-SUPPORT.

Table 250. Fields in the IPCONN report (continued)

Field Heading	Description
Link Security	The type of link authentication used: Certificate Securityname Source field: ISR-LINKAUTH
Receive Session Count	The number of receive sessions defined for this connection. Source field: ISR-RECEIVE-SESSIONS
Current Receive Session Count	The current number of receive sessions on this connection. Source field: ISR-CURRENT-RECEIVE-SESSIONS
Peak Receive Session Count	The peak number of receive sessions in use on this connection. Source field: ISR-PEAK-RECEIVE-SESSIONS
Total Allocates	The total number of allocate requests for this connection. Source field: ISR-TOTAL-ALLOCATES
Current Allocates Queued	The current number of allocate requests queued for this connection. Source field: ISR-CURRENT-QUEUED-ALLOCATES
Peak Allocates Queued	The peak number of allocate requests queued for this connection. Source field: ISR-PEAK-QUEUED-ALLOCATES
Allocates Failed - Link	The number of allocate requests that failed because the connection is released or out-of-service. Source field: ISR-ALLOCATES-FAILED-LINK
Allocates Failed - Other	The number of allocate requests that failed because a session is not currently available for use. Source field: ISR-ALLOCATES-FAILED-OTHER
Number of Transactions Attached	The total number of transactions that have been attached on this connection. Source field: ISR-TRANS-ATTACHED
Remote Terminal Starts	The total number of **START** requests sent from a remote terminal. Source field: ISR_REMOTE_TERM_STARTS
Transaction Routing Requests	The number of transaction routing requests sent across the connection. Source field: ISR-TR-REQUESTS
Transaction Routing Total Bytes Sent	The number of bytes sent by transaction routing requests. Source field: ISR-TR-BYTES-SENT
Transaction Routing Total Bytes Received	The number of bytes received on transaction routing requests. Source field: ISR-TR-BYTES-RECEIVED
Function Shipping Program requests	The number of program control requests function shipped across the connection. Source field: ISR-FS-PG-REQUESTS
Function Shipping Interval Control requests	The number of interval control requests function shipped across the connection. Source field: ISR-FS-IC-REQUESTS

Table 250. Fields in the IPCONN report (continued)

Field Heading	Description
Function Shipping Total requests	The total number of function shipping requests shipped across the connection. Source field: ISR-FS-PG-REQUESTS + ISR-FS-IC-REQUESTS + ISR-FS-FC-REQUESTS + ISR-FS-TD-REQUESTS + ISR-FS-TS-REQUESTS
Program Requests Total Bytes Sent	The number of bytes sent on program control requests. Source field: ISR-FS-PG-BYTES-SENT
Program Requests Total Bytes Received	The number of bytes received on program control requests. Source field: ISR-FS-PG-BYTES-RECEIVED
Interval Control Requests Total Bytes Sent	The number of bytes sent on interval control requests. Source field: ISR-FS-IC-BYTES-SENT
Interval Control Requests Total Bytes Received	The number of bytes received on interval control requests. Source field: ISR-FS-IC-BYTES-RECEIVED
IPCONN Network ID	The network ID of the remote system. Source field: ISR-NETWORK-ID
IPCONN Service Status	Whether data can be passed on the connection: Inservice Outservice Source field: **EXEC CICS INQUIRE IPCONN() SERVSTATUS()**
TCPIPSERVICE Name	The name of the PROTOCOL(IPIC) TCPIPSERVICE definition that defines the attributes of the inbound processing for this connection. Source field: ISR-TCPIP-SERVICE
User Authentication	The type of user authentication used: Defaultuser Identify Local Verify Source field: ISR-USERAUTH
Mirror Lifetime	The minimum lifetime of the mirror task for function shipped requests received by this region. The following options are included: REQUEST TASK UOW Source field: **EXEC CICS INQUIRE IPCONN() MIRRORLIFE()**
Send Session Count	The number of send sessions defined for this connection. Source field: ISR-SEND-SESSIONS
Current Send Session Count	The current number of send sessions on this connection. Source field: ISR-CURRENT-SEND-SESSIONS
Peak Send Session Count	The peak number of send sessions in use on this connection. Source field: ISR-PEAK-SEND-SESSIONS

Table 250. Fields in the IPCONN report (continued)

Field Heading	Description
Allocates per second	The number of allocate requests issued per second for this connection. Source field: ISR-TOTAL-ALLOCATES / Elapsed seconds since reset
Allocate Queue Limit	The maximum number of allocate requests that can be queued for this connection. Source field: ISR-ALLOCATE-QUEUE-LIMIT
Allocates Rejected - Queue Limit	The number of allocate requests that were rejected because the QUEUELIMIT value is reached. Source field: ISR-QLIMIT-ALLOC-REJECTS
Max Queue Time (seconds)	The maximum time, in seconds, for which allocate requests can be queued on this connection. Source field: ISR-MAX-QUEUE-TIME
Max Queue Time - Allocate Queue Purge	The number of times that the allocate request queue has been purged because the MAXQTIME value is reached. Source field: ISR-MAXQTIME-ALLOC-QPURGES
Max Queue Time - Allocates Purged	The total number of allocate requests purged because the queueing time exceeds the MAXQTIME value. Source field: ISR-MAXQTIME-ALLOCS-PURGED
XISQUE - Allocates Rejected	The number of allocate requests that were rejected by an XISQUE global user exit program. Source field: ISR-XISQUE-ALLOC-REJECTS
XISQUE - Allocate Queue Purge	The number of times that the allocate request queue has been purged by an XISQUE global user exit program. Source field: ISR-XISQUE-ALLOC-QPURGES
XISQUE - Allocates Purged	The total number of allocate requests purged because an XISQUE global user exit program requests that the queued allocate requests are purged. Source field: ISR-XISQUE-ALLOC-PURGED
Transaction Routing Average Bytes Sent	The average number of bytes sent by transaction routing requests. Source field: ISR-TR-BYTES-SENT / ISR-TR-REQUESTS
Program Requests Average Bytes Sent	The average number of bytes sent on program control requests. Source field: ISR-FS-PG-BYTES-SENT / ISR-FS-PG-REQUESTS
Interval Control Requests Average Bytes Sent	The average number of bytes sent on interval control requests. Source field: ISR-FS-IC-BYTES-SENT / ISR-FS-IC-REQUESTS
Function Shipping File Control requests	The number of file control requests for function shipping on this connection. Source field: ISR_FS_FC_REQUESTS
File Control Requests Total bytes sent	The number of bytes sent by file control requests. Source field: ISR_FS_FC_BYTES_SENT
File Control Requests Total Bytes Rcvd	The number of bytes received by file control requests. Source field: ISR_FS_FC_BYTES_RECEIVED

Table 250. Fields in the IPCONN report (continued)

Field Heading	Description
Function Shipping Temporary Storage Requests	The number of temporary storage requests for function shipping on this connection. Source field: ISR_FS_TS_REQUESTS
Temporary Storage Requests Total Bytes Sent	The number of bytes sent by temporary storage requests. Source field: ISR_FS_TS_BYTES_SENT
Temporary Storage Requests Total Bytes Rcvd	The number of bytes received by temporary storage requests. Source field: ISR_FS_TS_BYTES_RECEIVED
Function Shipping Transient Data Requests	The number of transient data requests for function shipping on this connection. Source field: ISR_FS_TD_REQUESTS
Transient Data Requests Total Bytes Sent	The number of bytes sent by transient data requests. Source field: ISR_FS_TD_BYTES_SENT
Transient Data Requests Total Bytes Rcvd	The number of bytes received by transient data requests. Source field: ISR_FS_TD_BYTES_RECEIVED
Unsupported Requests	The number of attempts to route requests for unsupported function across this connection. Source field: ISR_UNSUPPORTED_REQUESTS

Journalnames report

The Journalnames report is produced using a combination of the **EXEC CICS INQUIRE JOURNALNAME** and **EXEC CICS EXTRACT STATISTICS JOURNALNAME** commands. The statistics data is mapped by the **DFHLGRDS DSECT**.

Table 251. Fields in the Journalnames report

Field Heading	Description
Journal Name	The name of the journal. Source field: **EXEC CICS INQUIRE JOURNALNAME()**
Journal Status	The current journal status. Source field: **EXEC CICS INQUIRE JOURNALNAME() STATUS(cvda)**
Journal Type	The type of journal, MVS, SMF or Dummy. Source field: **EXEC CICS INQUIRE JOURNALNAME() TYPE(cvda)**
Logstream Name	The name of the logstream associated with this journal (MVS journals only). Source field: LGRSTREAM
Write Requests	The number of write requests for this journal. Source field: LGRWRITES
Bytes Written	The number of bytes written to this journal. Source field: LGRBYTES

Table 251. Fields in the Journalnames report (continued)

Field Heading	Description
Average Bytes	The average number of bytes written to this journal per request. Source field: (LGRBYTES / LGRWRITES)
Buffer Flushes	The number of buffer flush requests issued for this journal. Source field: LGRBUFLSH

JVM Programs report

The JVM Programs report shows information and statistics about Java programs that run in JVM servers or pooled JVMs. This report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** commands. The statistics data is mapped by the **DFHPGRDS DSECT**.

Table 252. Fields in the JVM Programs report

Field Heading	Description
Program Name	The name of the JVM program. Source field: **EXEC CICS INQUIRE PROGRAM()**
JVM server	The name of the JVMSERVER resource that the program requires to run in a JVM server, as specified in the JVMSERVER attribute of the PROGRAM resource. Source field: **EXEC CICS INQUIRE PROGRAM() JVMSERVER()**
Profile Name	The JVM profile that the program requires, as specified in the JVM attribute of the PROGRAM resource. Source field: **EXEC CICS INQUIRE PROGRAM() JVMPROFILE()**
Times Used	The number of times the program has been used. Source field: PGR-JVMPROGRAM-USECOUNT
EXEC Key	The execution key that the program requires, CICS key or user key, as specified in the EXECKEY attribute of the PROGRAM resource. Source field: **EXEC CICS INQUIRE PROGRAM() EXECKEY()**
JVMClass	The main class in the program, as specified in the JVMCLASS attribute of the PROGRAM resource. Source field: **EXEC CICS INQUIRE PROGRAM() JVMCLASS()**

JVMSERVERs report

The JVMSERVERs report shows information and statistics about JVMSERVER resource definitions. The JVMSERVER resource defines the runtime environment for a JVM server, including the JVM profile and the Language Environment runtime options.

This report is produced using a combination of **EXEC CICS INQUIRE JVMSERVER** and **EXEC CICS EXTRACT STATISTICS** commands. The statistics data is mapped by the DFHSJSDS DSECT.

Table 253. Fields in the JVMSERVERs report

Field Heading	Description
JVMSERVER Name	The name of the JVMSERVER resource definition. Source field: **EXEC CICS INQUIRE JVMSERVER**
JVMSERVER Enable Status	The status of the JVMSERVER resource definition. Source field: **EXEC CICS INQUIRE JVMSERVER () ENABLESTATUS**
JVMSERVER JVM profile name	The name of the JVM profile that is used to start the JVM server. Source field: SJS-JVMSERVER-JVMPROFILE
JVMSERVER LE runtime options	The name of the Language Environment runtime options program that is specified on the JVMSERVER resource. Source field: SJS-JVMSERVER-LE-RUNOPTS
JVMSERVER use count	The number of times that the JVM server has been called. Source field: SJS-JVMSERVER-USE-COUNT
JVMSERVER thread limit	The maximum number of threads in the JVM server. Source field: SJS-JVMSERVER-THREAD-LIMIT
JVMSERVER current threads	The current number of threads in the JVM server. Source field: SJS-JVMSERVER-THREAD-CURRENT
JVMSERVER peak threads	The peak number of threads in the JVM server. Source field: SJS-JVMSERVER-THREAD-HWM
JVMSERVER thread limit waits	The number of tasks that waited for a free thread. Source field: SJS-JVMSERVER-THREAD-WAITS
JVMSERVER thread limit wait time	The amount of time in seconds that tasks have waited for a free thread. Source field: SJS-JVMSERVER-THREAD-WAIT-TIME
JVMSERVER current thread waits	The number of tasks that are currently waiting for a free thread. Source field: SJS-JVMSERVER-THREAD-WAIT-CUR

Table 253. Fields in the JVMSERVERs report (continued)

Field Heading	Description
JVMSERVER peak thread waits	The peak number of threads that waited for a free thread. Source field: SJS-JVMSERVER-THREAD-WAIT-HWM
JVMSERVER system thread use count	The number of times that the system thread has been used. Source field: SJS-JVMSERVER-SYS-USE-COUNT
JVMSERVER system thread waits	The number of CICS tasks that waited for a system thread. Source field: SJS-JVMSERVER-SYS-WAITED
JVMSERVER system thread wait time	The accumulated time in seconds that tasks spent waiting for a system thread. Source field: SJS-JVMSERVER-SYS-WAITED-TIME
JVMSERVER current sys thread waits	The current number of tasks that are waiting for a system thread. Source field: SJS-JVMSERVER-SYS-WAIT-CUR
JVMSERVER peak system thread waits	The highest number of tasks that waited for a system thread. Source field: SJS-JVMSERVER-SYS-WAIT-HWM
JVMSERVER current heap size	The size in bytes of the heap that is currently allocated to the JVM server. Source field: SJS-JVMSERVER-MAX-HEAP
JVMSERVER initial heap size	The size in bytes of the initial heap that is allocated to the JVM server. This value is set by the **-Xms** option in the JVM profile. Source field: SJS-JVMSERVER-CURRENT-HEAP
JVMSERVER maximum heap size	The size in bytes of the maximum heap that can be allocated to the JVM server. This value is set by the **-Xmx** option in the JVM profile. Source field: SJS-JVMSERVER-INITIAL-HEAP
JVMSERVER peak heap size	The size in bytes of the largest heap that has been allocated to the JVM server. Source field: SJS-JVMSERVER-PEAK-HEAP
JVMSERVER heap occupancy	The size in bytes of the heap immediately after the last garbage collection occurred. Source field: SJS-JVMSERVER-OCCUPANCY

Table 253. Fields in the JVMSERVERs report (continued)

Field Heading	Description
JVMSERVER Garbage Collection (GC)	The garbage collection policy that is being used by the JVM. Source field: SJS-JVMSERVER-GC-POLICY
JVMSERVER no. of major GC events	The number of major garbage collection events that have occurred. Source field: SJS-JVMSERVER-MJR-GC-EVENTS
JVMSERVER total elapsed time spent in major GC	The total elapsed time in milliseconds that was spent performing major garbage collection. Source field: SJS-JVMSERVER-MJR-GC-CPU
JVMSERVER total memory freed by major GC	The total memory in bytes that was freed by performing major garbage collection. Source field: SJS-JVMSERVER-MJR-HEAP-FREED
JVMSERVER no. of minor GC events	The number of minor garbage collections that have occurred. Source field: SJS-JVMSERVER-MNR-GC-EVENTS
JVMSERVER total elapsed time spent in minor GC	The total elapsed time in milliseconds that was spent performing minor garbage collection. Source field: SJS-JVMSERVER-MNR-GC-CPU
JVMSERVER total memory freed by minor GC	The total memory in bytes that was freed by performing minor garbage collection. Source field: SJS-JVMSERVER-MNR-HEAP-FREED

LIBRARY reports

There are two LIBRARY reports, LIBRARYs report, and LIBRARY Data set Concatenation report.

LIBRARYs report

The LIBRARYs report is produced using a combination of **EXEC CICS INQUIRE** LIBRARY and **EXEC CICS EXTRACT STATISTICS** LIBRARY RESID commands. The statistics data is mapped by the DFHLDBDS DSECT.

Table 254. Fields in the LIBRARYs report

Field Heading	Description
LIBRARY Name	The name of the LIBRARY. Source field: **EXEC CICS INQUIRE LIBRARY**
Search Position	The current absolute position of this LIBRARY in the overall LIBRARY search order. Source field: **EXEC CICS INQUIRE LIRBARY SEARCHPOS**

Table 254. Fields in the LIBRARYs report (continued)

Field Heading	Description
Ranking	The position this LIBRARY appears in the overall LIBRARY search order relative to other LIBRARY concatenations. Source field: **EXEC CICS INQUIRE LIBRARY RANKING**
Critical	Indicates whether this LIBRARY is critical to CICS startup. Source field: **EXEC CICS INQUIRE LIBRARY CRITICAL**
Enable Status	Indicates whether the LIBRARY is included in the overall LIBRARY search order. Source field: **EXEC CICS INQUIRE LIBRARY ENABLESTATUS**
Program Loads	The number of times the loader has issued an MVS LOAD request to load programs from the LIBRARY concatenation into CICS managed storage. Source field: LDB-LIBRARY-PROG-LOADS
Number Dsnames	The number of data sets in the LIBRARY concatenation. Source field: **EXEC CICS LIBRARY NUMDSNAMES**
Concatenation	The concatenation number of the data set in the LIBRARY concatenation. Source field: **EXEC CICS INQUIRE LIBRARY DSNAME01-16**
Data set Name	The 44 character name of each data set in the LIBRARY concatenation. Source field: **EXEC CICS INQUIRE LIBRARY DSNAME01-16**
Dsname Number	The position that the data set occupies within the LIBRARY. **Note:** DFHRPL does not have any Dsname Numbers.

LIBRARY Data set Concatenation report

The LIBRARY Data set Concatenation report is produced using a combination of **EXEC CICS INQUIRE LIBRARY** and **EXEC CICS EXTRACT STATISTICS LIBRARY RESID()** commands.

The statistics data is mapped by the DFHLDBDS DSECT.

Table 255. Fields in the LIBRARY Data set Concatenation report

Field Heading	Description
Concatenation	The concatenation number of the data set based on a concatenation of all LIBRARYs in the search order in which they appear. Source field: Generated by DFH0STAT
Dataset Name	The 44 character name of each data set in the LIBRARY concatenation. Source field: **EXEC CICS INQUIRE LIBRARY DSNAME01-16**
Dsname Number	The position that the data set occupies within the LIBRARY. **Note:** DFHRPL does not have any Dsname Numbers. Source field: Generated by DFH0STAT
LIBRARY Name	The name of the LIBRARY. Source field: **EXEC CICS INQUIRE LIBRARY**

Table 255. Fields in the LIBRARY Data set Concatenation report (continued)

Field Heading	Description
Ranking	The position this LIBRARY appears in the overall LIBRARY search order relative to other LIBRARY concatenations. Source field: **EXEC CICS INQUIRE LIBRARY RANKING**
Critical	Indicates whether this LIBRARY is critical to CICS startup. Source field: **EXEC CICS INQUIRE LIBRARY CRITICAL**

Loader and Program Storage report

The Loader and Program Storage report is produced using a combination of the **EXEC CICS EXTRACT STATISTICS PROGRAM** and **EXEC CICS EXTRACT STATISTICS STORAGE** commands. The statistics data is mapped by the **DFHLDGDS** and **DFHSMDDS DSECT**.

Table 256. Fields in the Loader Storage report

Field Heading	Description
LIBRARY Load requests	The number of times the loader issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Modules in the LPA are not included in this value. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. Source field: LDGLLR
LIBRARY Load Rate per second	The number of times per second the loader has issued an MVS LOAD request to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Modules in the LPA are not included in this value. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. Source field: LDGLLR / Elapsed seconds (since the last statistics reset)
LIBRARY Load requests on the RO TCB	The number of times the loader issued a program load request that used the RO (resource-owning) TCB. This value is a subset of the number of library loads shown by “LIBRARY Load requests”. To calculate the number of program load requests that ran on open TCBs, subtract this value from the value shown by “LIBRARY Load requests”. Source field: LDGLLRRO
Total LIBRARY Load time	The total time taken to load programs from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Modules in the LPA are not included in this value. The value includes both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. Source field: LDGLLT
Total LIBRARY Load time on the RO TCB	The total time taken for program load requests that ran on the RO TCB. This value is a subset of the number of library loads shown by “Total LIBRARY Load time”. To calculate the number of program load requests that ran on open TCBs, subtract this value from the value shown by “Total LIBRARY Load time”. Source field: LDGLLTRO

Table 256. Fields in the Loader Storage report (continued)

Field Heading	Description
Average LIBRARY Load time	The average time taken to load a program. The value is an average including both program load requests that ran on open TCBs, and program load requests that used the RO (resource-owning) TCB. Source field: (LDGLLT / LDGLLR)
Total Program Uses	The number of uses of any program by the CICS system. Source field: LDGPUSES
Average LIBRARY Load time on the RO TCB	The average time taken to complete only those program load requests that used the RO (resource-owning) TCB. Source field: (LDGLLT / LDGLLR)
Program Use to Load Ratio	The ratio of program uses to programs loads. Source field: (LDGPUSES / LDGLLR)
LIBRARY Load requests that waited	The number of loader domain requests that waited for the loader domain to complete an operation on the program on behalf of another task. This figure is the number of tasks that waited in the past, and does not include tasks that are currently waiting ("Current Waiting LIBRARY Load requests"). Program load requests might wait for the following reasons: • The program is being loaded by another task that is running on an open TCB. • The loader domain is searching the link pack area (LPA) for the program. • A NEWCOPY request or physical load is in progress for the program. Source field: LDGWTDLR
Total LIBRARY Load request wait time	The total suspended time for the number of tasks shown in "LIBRARY Load requests that waited". Source field: LDGTTW
Times LIBRARY secondary extents detected	The number of times the loader received an end-of-extent condition during a LOAD and successfully closed and reopened the DFHRPL or dynamic LIBRARY and retried the LOAD. Source field: LDGDREBS
Average LIBRARY Load request wait time	The average loader domain request suspend time. Source field: (LDGTTW / LDGWTDLR)
Current Waiting LIBRARY Load requests	The number of loader domain requests that are currently waiting for the loader domain to complete an operation on the program on behalf of another task. Program load requests might wait for the following reasons: • The program is being loaded by another task that is running on an open TCB. • The loader domain is searching the link pack area (LPA) for the program. • A NEWCOPY request or physical load is in progress for the program. Source field: LDGWLR
Peak Waiting LIBRARY Load requests	The maximum number of tasks suspended at one time. Source field: LDGWLRHW

Table 256. Fields in the Loader Storage report (continued)

Field Heading	Description
Times at Peak	The number of times the high-water mark shown in “Peak Waiting LIBRARY Load requests” was reached. This value together with the previous two values indicate the level of contention for loader resource. Source field: LDGHWMT
Average Not-In-Use program size	The average size of a program currently on the Not-In-Use queue. Source field: ((LDGSTGNIU + LDGSNIU + LDGRECNIU + LDGECNIU + LDGESNIU + LDGERNIU) / 1024) / LDGPROGNIU)
Programs Removed by compression	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	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	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	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	The number of programs on the Not-In-Use (NIU) queue. Source field: LDGPROGNIU
LIBRARY search order updates	The number of updates to the LIBRARY search order. Source field: LDGLBSOU
Total LIBRARY search order update time	The total time spent updating the LIBRARY search order. Source field: LDGLSORT
Average LIBRARY search order update time	The average time spent updating the LIBRARY search order. Source field: LDGLSORT/LDGLBSOU
Load requests waited - search order update	The total number of waits for programs to load while the search order is being updated. These operations could be: • Install of a dynamic LIBRARY. • Enable or disable of a dynamic LIBRARY. • Change in RANKING of a dynamic LIBRARY. Source field: LDGLWSOU

Table 257. Fields in the Program Storage report

Field Heading	Description
Nucleus Program Storage (CDSA)	The current amount of storage allocated to nucleus programs in the CDSA. Source field: (SMDCPS for subpool 'LDNUC ' and 'LDNRS / 1024)
Nucleus Program Storage (ECDSA)	The current amount of storage allocated to nucleus programs in the ECDSA. Source field: (SMDCPS for subpool 'LDENUC ' and 'LDENRS ' / 1024)
Program Storage (SDSA)	The current amount of storage allocated to programs in the SDSA. Source field: (SMDCPS for subpool 'LDPGM ' / 1024)
Program Storage (ESDSA)	The current amount of storage allocated to programs in the ESDSA. Source field: (SMDCPS for subpool 'LDEPGM ' / 1024)
Resident Program Storage (SDSA)	The current amount of storage allocated to resident programs in the SDSA. Source field: (SMDCPS for subpool 'LDRES ' / 1024)
Resident Program Storage (ESDSA)	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)	The current amount of storage allocated to nucleus programs in the RDSA. Source field: (SMDCPS for subpool 'LDNUCRO ' and 'LDNRSRO ' / 1024)
Read-Only Nucleus Program Storage (ERDSA)	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)	The current amount of storage allocated to programs in the RDSA. Source field: (SMDCPS for subpool 'LDPGMRO ' / 1024)
Read-Only Program Storage (ERDSA)	The current amount of storage allocated to programs in the ERDSA. Source field: (SMDCPS for subpool 'LDEPGMRO ' / 1024)
Read-Only Resident Program Storage (RDSA)	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)	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	The current amount of CDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(1) / 1024)
ECDSA used by Not-In-Use programs	The current amount of ECDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(2) / 1024)
SDSA used by Not-In-Use programs	The current amount of SDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(3) / 1024)
ESDSA used by Not-In-Use programs	The current amount of ESDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(4) / 1024)

Table 257. Fields in the Program Storage report (continued)

Field Heading	Description
RDSA used by Not-In-Use programs	The current amount of RDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(5) / 1024)
ERDSA used by Not-In-Use programs	The current amount of ERDSA storage that is occupied by Not-In-Use (NIU) programs. Source field: (LDGSTGNIU(6) / 1024)

Logstreams reports

Four Logstream reports are produced using the **EXEC CICS EXTRACT STATISTICS** STREAMNAME and **EXEC CICS INQUIRE** STREAMNAME commands. The statistics data is mapped by the **DFHLGGDS DSECT**.

For more information about logstreams, see Chapter 16, "CICS logging and journaling: Performance and tuning," on page 225.

Table 258. Fields in the Logstream Global report

Field Heading	Description
Activity Keypoint Frequency (AKPFREQ)	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. Source field: **EXEC CICS INQUIRE STREAMNAME**
Activity Keypoints Taken	The number of activity keypoints taken. Source field: **EXEC CICS INQUIRE STREAMNAME()**
Average time between Activity Keypoints	The average time between the taking of activity keypoints.
Logstream Deferred Force Interval (LGDFINT)	The current logstream deferred force interval. Source field: **EXEC CICS INQUIRE STREAMNAME**

The Logstream System Logs Report is produced using the **EXEC CICS INQUIRE STREAMNAME** and **EXEC CICS EXTRACT STATISTICS STREAMNAME** commands. The statistics data is mapped by the **DFHLGSDS DSECT**.

Table 259. Fields in the Logstream System Logs report

Field Heading	Description
Logstream Name	The name of the logstream. Source field: **EXEC CICS INQUIRE STREAMNAME()**
Logstream Status	The current status of the logstream. Source field: **EXEC CICS INQUIRE STREAMNAME() STATUS()**
DASD Only	Indicates the type of logstream. If set to YES, the logstream is of type DASDONLY. If set to NO, the log stream is of type coupling facility (CF). Source field: LGSDONLY

Table 259. Fields in the Logstream System Logs report (continued)

Field Heading	Description
Retention Period (days)	The logstream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Source field: LGSRETPD
Coupling Facility Structure Name	The coupling facility (CF) structure name for the logstream. The structure name is only applicable to coupling facility type logstreams. Source field: LGSSTRUC
Auto Delete	The log data auto delete indicator. If set to YES the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to NO, the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Source field: LGSAUTOD
Logstream Writes	The number of write (IXGWRITE) requests issued to this logstream. Source field: LGSWRITES
Maximum Block Length	The maximum block size allowed by the MVS Logger for the logstream. Source field: LGSMAXBL
Logstream Writes per second	The number of logstream writes per second for this logstream. Source field: (LGSWRITES / ELAPSED-SECONDS)
Average Bytes per Logstream Write	The average number of bytes written to this logstream per write request. Source field: (LGSBYTES / LGSWRITES)
Logstream Deletes (Tail Trims)	The number of delete (IXGDELET) requests issued to this logstream. Source field: LGSDELETES
Logstream Query Requests	The number of query requests issued for this logstream. Source field: LGSQUERIES
Logstream Browse Starts	The number of browse start requests issued for this logstream. Source field: LGSBRWSTRT
Logstream Browse Reads	The number of browse read requests issued for this logstream. Source field: LGSBRWREAD
Logstream Buffer Appends	The number of occasions on which a journal record was successfully appended to the current log stream buffer. Source field: LGSBUFAPP
Logstream Buffer Full Waits	The number of times buffer full has occurred for this logstream. Source field: LGSBUFWAIT
Logstream Force Waits	The total number of tasks suspending while requesting a flush of the logstream buffer currently in use. Source field: LGSTFCWAIT
Logstream Current® Force Waiters	The current number of force waiters for this logstream. Source field:

Table 259. Fields in the Logstream System Logs report (continued)

Field Heading	Description
Logstream Retry Errors	The number of occasions on which MVS system logger retryable errors occurred when a block of data was being written to the log stream. Source field: LGSRTYERRS
Logstream Peak Force Waiters	The peak number of force waiters for this logstream. Source field: LGSPKFWTRS

The Logstreams Resource Report is produced using the **EXEC CICS INQUIRE STREAMNAME** and **EXEC CICS EXTRACT STATISTICS STREAMNAME** commands. The statistics data is mapped by the **DFHLGSDS DSECT**.

Table 260. Fields in the Logstreams Resource report

Field Heading	Description
Logstream Name	The name of the logstream. Source field: **EXEC CICS INQUIRE STREAMNAME()**
Use Count	The current use count of the logstream. Source field: **EXEC CICS INQUIRE STREAMNAME() USECOUNT()**
Status	The current status of the logstream. Source field: **EXEC CICS INQUIRE STREAMNAME() STATUS()**
Sys Log	Indicates if the log stream forms part of the System Log. Source field: LGSSYSLG
Structure Name	The coupling facility (CF) structure name for the log stream. The structure name is only applicable to coupling facility type logstreams. Source field: LGSSTRUC
Max Block Length	The maximum block size allowed by the MVS Logger for the log stream. Source field: LGSMAXBL
DASD Only	Indicates the type of log stream. If set to YES the log stream is of type DASDONLY. If set to NO the log stream is of type coupling facility (CF). Source field: LGSDONLY
Retention Period	The log stream retention period (in days) that the data must be kept before it can be physically deleted by the MVS Logger. Source field: LGSRETPD
Auto Delete	The log data auto delete indicator. If set to YES the MVS Logger automatically deletes the data as it matures beyond the retention period, irrespective of any logstream delete calls. If set to NO the data is only deleted when a logstream delete call is issued and the data has matured beyond the retention period. Source field: LGSAUTOD
Stream Deletes	The number of delete (IXGDELET) requests issued for this logstream. Source field: LGSDELETES
Browse Starts	The number of browse start requests issued for this logstream. Source field: LGSBRWSTRT

Table 260. Fields in the Logstreams Resource report (continued)

Field Heading	Description
Browse Reads	The number of browse read requests issued for this logstream. Source field: LGSBRWREAD

The Logstreams Requests Report is produced using the **EXEC CICS INQUIRE STREAMNAME** and**EXEC CICS EXTRACT STATISTICS STREAMNAME** commands. The statistics data is mapped by the **DFHLGSDS DSECT**.

Table 261. Fields in the Logstreams Requests report

Field Heading	Description
Logstream Name	The name of the logstream. Source field:**EXEC CICS INQUIRE STREAMNAME()**
Write Requests	The number of IXGWRITE requests issued to this logstream. IXGWRITE occurs, for example, when the logstream buffer is full, or when the application issues an **EXEC CICS WRITE JOURNALNAME** command with the WAIT option specified. Source field: LGSWRITES
Bytes Written	The number of bytes written to this logstream. Source field: LGSBYTES
Average Bytes	The average number of bytes written to this logstream per request. Source field: (LGSBYTES / LGSWRITES)
Buffer Appends	The number of occasions on which a journal record was successfully appended to the current logstream buffer. Source field: LGSBUFAPP
Buffer Full Waits	The number of times buffer full has occurred for this logstream. Source field: LGSBUFWAIT
Force Waits	The total number of force waits for this logstream. Source field: LGSTFCWAIT
Peak Waiters	The peak number of force waiters for this logstream. Source field: LGSPKFWTRS
Retry Errors	The number of occasions on which MVS logger retry errors occurred when a block of data was being written to the log stream. Source field: LGSRTYERRS

LSR pools report

The LSR pools report is produced using the **EXEC CICS EXTRACT STATISTICS LSRPOOL** command. The statistics data is mapped by the **DFHA08DS DSECT**.

If you have combined data and index buffers, the report presents the statistics for data buffers and index buffers together as “Data and Index Buffer Statistics”. If you have separate data and index buffers, the report presents the statistics separately, as “Data Buffer Statistics” and “Index Buffer Statistics”.

Table 262. Fields in the LSR pools report

Field Heading	Description
Pool Number	The identifying number of the LSR pool. This value must be in the range 1 - 255.
Time Created	The time when this LSR pool was created. Source field: A08LBKCD
Maximum key length	The length of the largest key of a VSAM data set that can use this LSR pool. Source field: A08BKKYL
Total number of strings	The total number of VSAM strings defined for this LSR pool. Source field: A08BKSTN
Peak concurrently active strings	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 higher 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	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	The highest number of requests that were queued at one time because all the strings in the pool were in use. Source field: A08BKHSW
Data Buffers	The number of data buffers specified for the LSR pool. Source field: A08TDBFN
Hiperspace Data Buffers	The number of Hiperspace data buffers specified for the LSR pool. Source field: A08TDHBW
Successful look asides	The number of successful lookasides to data buffers for this LSR pool. Source field: A08TDBFF
Buffer reads	The number of read I/O operations to the data buffers for this LSR pool. Source field: A08TDFRD
User initiated writes	The number of user-initiated I/O writes from the data buffers for this LSR pool. Source field: A08TDUIW
Non-user initiated writes	The number of non-user-initiated I/O writes from the data buffers for this LSR pool. Source field: A08TDNUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace data buffers to virtual data buffers. Source field: A08TDCRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual data buffers to Hiperspace data buffers. Source field: A08TDCWS

Table 262. Fields in the LSR pools report (continued)

Field Heading	Description
Failing Hiperspace CREADS	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	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	The number of index buffers specified for the LSR pool. Source field: A08TIBFN
Hiperspace Index Buffers	The number of Hiperspace index buffers specified for the LSR pool. Source field: A08TIHBW
Successful look asides	The number of successful lookasides to index buffers for this LSR pool. Source field: A08TIBFF
Buffer reads	The number of read I/Os to the index buffers for this LSR pool. Source field: A08TIFRD
User initiated writes	The number of user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TIUIW
Non-user initiated writes	The number of non-user-initiated buffer writes from the index buffers for this LSR pool. Source field: A08TINUW
Successful Hiperspace CREADS	The number of successful CREAD requests issued to transfer data from Hiperspace index buffers to virtual index buffers. Source field: A08TICRS
Successful Hiperspace CWRITES	The number of successful CWRITE requests issued to transfer data from virtual index buffers to Hiperspace index buffers. Source field: A08TICWS
Failing Hiperspace CREADS	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	The number of CWRITE requests that failed. There was insufficient Hiperspace and VSAM had to write the index data to DASD. Source field: A08TICWF
Buffer Size	The size of the data buffers that are available to CICS. Source field: A08BKBSZ
No. of Buffers	The number of buffers of each size available to CICS. Source field: A08BKBFN
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08BKHBN

Table 262. Fields in the LSR pools report (continued)

Field Heading	Description
Look Asides	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
Buffer Reads	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	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	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
Look-Aside Ratio	The ratio of buffer lookasides to buffer reads. Source field:((A08BKBFF / (A08BKBFF + A08BKFRD)) * 100)
Successful CREADS/ CWRITES	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers, and of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08BKCRS + A08BKCWS
Failing CREADS/ CWRITES	The number of CREAD requests that failed (because MVS had withdrawn the space and VSAM had to read data from DASD), and the number of CWRITE requests that failed (because there was insufficient Hiperspace and VSAM had to write the data to DASD). Source field: A08BKCRF + A08BKCWF
Buffer Size	The size of the index data buffers that are available to CICS. Source field: A08IKBSZ

Table 262. Fields in the LSR pools report (continued)

Field Heading	Description
No. of Buffers	The number of buffers of each size available to CICS. Source field: A08IKBFN
Hiperspace Buffers	The number of Hiperspace buffers specified for the pool. Source field: A08IKHBN
Look Asides	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
Buffer Reads	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	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	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
Look-Aside Ratio	The ratio of buffer look asides to buffer reads. Source field: ((A08BKBFF / (A08BKBFF + A08BKFRD)) * 100)
Successful CREADS/ CWRITES	The number of successful CREAD requests issued to transfer data from Hiperspace buffers to virtual buffers, and of successful CWRITE requests issued to transfer data from virtual buffers to Hiperspace buffers. Source field: A08IKCRS + A08IKCWS

Table 262. Fields in the LSR pools report (continued)

Field Heading	Description
Failing CREADS/ CWRITES	The number of CREAD requests that failed (because MVS had withdrawn the space and VSAM had to read data from DASD), and the number of CWRITE requests that failed (because there was insufficient Hiperspace and VSAM had to write the data to DASD). Source field: A08IKCRF + A08IKCWF

Page Index report

The Page Index report contains a complete list of all the statistics reports produced by **DFH0STAT**, and shows the first page number for each statistics report.

PIPELINEs report

The PIPELINEs report is produced using a combination of **EXEC CICS INQUIRE PIPELINE** and **EXEC CICS EXTRACT STATISTICS PIPELINE RESID()** commands. The statistics data is mapped by the **DFHPIRDS DSECT**.

Table 263. Fields in the PIPELINEs report

Field Heading	Description
PIPELINE Name	The name of the PIPELINE resource definition. Source field: **EXEC CICS INQUIRE PIPELINE**
PIPELINE Mode	The operating mode of the pipeline. Source field: **EXEC CICS INQUIRE PIPELINE() MODE()**
PIPELINE Message Format	The message format processed by the PIPELINE. Source field: **EXEC CICS INQUIRE PIPELINE() MSGFORMAT()**
PIPELINE Enable Status	Whether the PIPELINE definition is enabled or disabled. Source field: **EXEC CICS INQUIRE PIPELINE() ENABLESTATUS**
Configuration file	The name of the z/OS UNIX file that provides information about the message handlers and their configuration. Source field: **EXEC CICS INQUIRE PIPELINE() CONFIGFILE**
Shelf directory	The fully qualified name of the shelf directory for the PIPELINE definition. Source field: **EXEC CICS INQUIRE PIPELINE() SHELF**
WSDIR pickup directory	The fully qualified name of the Web service binding directory (also known as the pickup directory). Source field: **EXEC CICS INQUIRE PIPELINE() WSDIR**
PIPELINE use count	The number of times this PIPELINE resource definition was used to install a web service or to process a web service request. Source field: PIR-PIPELINE-USE-COUNT
JSON JAVA parser	For a JSON PIPELINE resource, specifies if the JSON request message is parsed by using Java or from within the CICS pipeline. Source field: PIR_JSON_JAVA_PARSER

Program reports

There are five program reports, Programs report, Programs definitions report, Program Autoinstall report, Programs by DSA and LPA report, and Program Totals report.

Programs report

The Programs report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

Information about Java programs that run in a JVM is handled differently from information about other programs, because JVM programs are not loaded by CICS. For JVM programs, the Programs Report shows only the program name, execution key, and use count. This information is obtained using the **EXEC CICS EXTRACT STATISTICS JVMPROGRAM** command. For full information about JVM programs, see "JVM Programs report" on page 842.

Table 264. Fields in the Programs report

Field Heading	Description
Program Name	The name of the program. Source field: **EXEC CICS INQUIRE PROGRAM**
Data Loc	The storage location that the program is able to accept. Source field: **EXEC CICS INQUIRE PROGRAM DATALOCATION**
Exec Key	The access key in which the program will execute. Source field: **EXEC CICS INQUIRE PROGRAM EXECKEY**
Times Used	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	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)
LIBRARY name	The name of the LIBRARY from which the program was just loaded (non-LPA resident modules only). Source field: LDRLBNM

Table 264. Fields in the Programs report (continued)

Field Heading	Description
LIBRARY Offset	The offset into the DFHRPL or dynamic LIBRARY concatenation of the data set from which the program was last loaded (non-LPA resident modules only). If this field is blank, it indicates that the program has never been loaded, or that it has not been loaded from the LIBRARY. A value of zero appearing in the report indicates that the program has been loaded at least once from the LIBRARY, and has an offset value of zero. Source field: LDRRPLO
Times Newcopy	The number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	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	The size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE
Program Location	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

Programs definitions report

The programs definitions report is produced by using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAMDEF** commands. The statistics data was mapped by the **DFHPGDDS DSECT**.

Table 265. Fields in the programs definitions report

Field Heading	Description
Program Name	The name of the program. Source field: **EXEC CICS INQUIRE PROGRAM**
Type	The type of module. Source field: PGD-PROGRAM-TYPE
Concurrency	The concurrency attribute (QUASIRENT, THREADSAFE, or REQUIRED) of the installed program definition. Source field: PGD-PROGRAM-CONCURRENCY

Table 265. Fields in the programs definitions report (continued)

Field Heading	Description
API	The API attribute (CICS or OPEN) of the installed program definition. Source field: PGD-PROGRAM-API
EXEC key	The access key in which the program runs. Source field: PGD-PROGRAM-EXEC-KEY
Data Location	The storage location that the program is able to accept. Source field: PGD-PROGRAM-DATA-LOC
Language Deduced	The language of the module. Source field: PGD-PROGRAM-LANG-DEDUCED
Runtime Environment	The runtime environment of the program. Source field: PGD-PROGRAM-RUNTIME-ENV
JVM server	For a Java program, the name of the JVM server in which this Java program runs. Source field: PGD-PROGRAM-JVMSERVER
Dynamic	Whether, if the program is the subject of a program-link request, it can be dynamically routed. Source field: PGD-PROGRAM-DYNAMIC
Remote Name	For programs only, the name by which the module is known in the CICS region that is named in the **Remote System** field, and only to those programs defined to be remote. Source field: PGD-PROGRAM-REMOTE-NAME
Remote Tran	For programs only, the name of the transaction under which this module, which must be a program, runs remotely. The transaction identifier that the remote region assigns to the task created there to run it when a task in the local region LINKs to it. Source field: PGD-PROGRAM-TRAN-ID
Remote System	For programs only, the name of the CICS region in which the module is defined. It applies only to programs, and only to those programs defined to be remote. Source field: PGD-PROGRAM-REMOTE-SYSID

Program Autoinstall report

The Program Autoinstall report shows information and statistics about the status of program autoinstall, catalog program definitions, and the number of autoinstalls that were attempted, rejected, and failed.

The Program Autoinstall report is produced using a combination of the **EXEC CICS INQUIRE SYSTEM**, and the **EXEC CICS COLLECT STATISTICS PROGAUTO** commands. The statistics data is mapped by the DFHPGGDS DSECTs.

Table 266. Fields in the Program Autoinstall report

Field Heading	Description
Program Autoinstall Status	Indicates the current status of program autoinstall. Source field: **EXEC CICS INQUIRE SYSTEM PROGAUTOINST(cvda)**

Table 266. Fields in the Program Autoinstall report (continued)

Field Heading	Description
Autoinstall Program	The name of the user-replaceable program autoinstall model definition program. Source field: **EXEC CICS INQUIRE SYSTEM PROGAUTOEXIT()**
Catalog Program Definitions	Indicates whether, and when, autoinstalled program definitions are to be cataloged. Source field: **EXEC CICS INQUIRE SYSTEM PROGAUTOCTLG(cvda)**
Autoinstalls attempted	The number of program autoinstalls attempted. Source field: PGGATT
Autoinstalls rejected	The number of program autoinstalls rejected by the program autoinstall user-replaceable program. Source field: PGGREJ
Autoinstalls failed	The number of program autoinstalls failed for reasons other than being rejected by the program autoinstall user-replaceable program. Source field: PGGFAIL

Programs by DSA and LPA report

The Programs by DSA and LPA report is produced using a combination of the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

Table 267. Fields in the Programs by DSA and LPA report

Field Heading	Description
Program Name	The name of the program. Source field: **EXEC CICS INQUIRE PROGRAM()**
Concurrency Status	The concurrency attribute of the program (QUASIRENT, THREADSAFE, or REQUIRED). Source field: **EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda**)
API Status	The API attribute of the program (CICS or open API). Source field:**EXEC CICS INQUIRE PROGRAM() APIST(cvda)**
Times Used	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 can cause the loader domain to issue an MVS LOAD. Source field: LDRTU
Times Fetched	The number of times the loader domain has issued an MVS LOAD request to load a copy of the program from the static DFHRPL or dynamic LIBRARY concatenation into CICS managed storage. Source field: LDRFC
Total Fetch Time	The time taken to perform all fetches for this program. Source field: LDRFT
Average Fetch Time	The average time taken to perform a fetch of the program. Source field: (LDRFT / LDRFC)

Table 267. Fields in the Programs by DSA and LPA report (continued)

Field Heading	Description
LibDsn Offset	The offset into the LIBRARY DD concatenation from which the program was last loaded (non-LPA resident modules only). If this field is blank, it indicates that the program has never been loaded, or that it has not been loaded from the LIBRARY. A value of zero appearing in the report indicates that the program has been loaded at least once from the LIBRARY, and has an offset value of zero. Source field: LDRRPLO
Times Newcopy	The number of times a NEWCOPY has been requested against this program. Source field: LDRTN
Times Removed	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	The size of the program in bytes, if known (otherwise zero). Source field: LDRPSIZE
Program Location	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

The Program Totals Report is calculated from data obtained using the **EXEC CICS INQUIRE PROGRAM** and **EXEC CICS EXTRACT STATISTICS PROGRAM** commands. The statistics data was mapped by the **DFHLDRDS DSECT**.

Information about Java programs that run in a JVM is handled differently from information about other programs, because these programs are not loaded by CICS. The number of Java programs that run in a JVM is included in the Program Totals Report. For full information about JVM programs, see "JVM Programs report" on page 842.

Table 268. Fields in the Program Totals report

Field Heading	Description
Programs	The current total number of programs defined to CICS in all languages. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - Assembler	The current total number of programs defined to CICS as Assembler programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.

Table 268. Fields in the Program Totals report (continued)

Field Heading	Description
Programs - C	The current total number of programs defined to CICS as C programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - COBOL	The current total number of programs defined to CICS as COBOL programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - Java (JVM)	The current total number of programs defined to CICS as Java programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - Language Environment	The current total number of programs defined to CICS as Language Environment programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - PL1	The current total number of programs defined to CICS as PL/I programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - Remote	The current total number of programs defined to CICS as remote programs. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Programs - Not Deduced	The current total number of programs defined to CICS but whose language was not specified in the resource definition. Source field: **EXEC CICS INQUIRE PROGRAM LANGDEDUCED(cvda) RUNTIME(cvda)**.
Maps	The current number of maps defined to CICS.
Partitionsets	The current number of partitionsets defined to CICS.
Total	The total number of programs, maps, and partitionsets defined to CICS.
CDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the CDSA.
SDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the SDSA.
RDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the RDSA.
ECDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ECDSA.
ESDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ESDSA.
ERDSA Programs	The number of programs, maps, and partitionsets defined to CICS currently residing in the ERDSA.
LPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the LPA.
ELPA Programs	The current number of programs, maps, and partitionsets defined to CICS residing in the ELPA.
Unused Programs	The current number of programs, maps, and partitionsets defined to CICS and which have been located in DFHRPL or a dynamic LIBRARY concatenation but which have not been used by any CICS task.
Not Located Programs	The current number of programs, maps, and partitionsets defined to CICS but which have not been located in any DFHRPL or a dynamic LIBRARY concatenation.
Total	The total number of programs, maps, and partitionsets defined to CICS.

Recovery Manager report

The Recovery Manager report is produced using the **EXEC CICS EXTRACT STATISTICS RECOVERY** command. The statistics data is mapped by the **DFHRMGDS DSECT**.

Table 269. Fields in the Recovery Manager report

Field Heading	Description
Number of Syncpoints forward	The number of syncpoints issued. Source field: RMGSYFWD
Number of Syncpoints backward	The number of syncpoint rollbacks issued. Source field: RMGSYBWD
Number of Resynchronizations	The number of resyncs issued. Source field: RMGRESYN
Total UOWs shunted for indoubt failure	The total number of UOWs shunted for indoubt failure. Source field: RMGTSHIN
Total time UOWs shunted for indoubt failure	The total time UOWs were shunted for indoubt failure. Source field: RMGTSHTI
Current UOWs shunted for indoubt failure	The current number of UOWs shunted for indoubt failure. Source field: RMGCSHIN
Total time current UOWs shunted for indoubt failure	The total time for the current UOWs shunted for indoubt failure. Source field: RMGCSHTI
Total UOWs shunted for commit/backout failure	The total number of UOWs shunted for commit/backout failure. Source field: RMGTSHRO
Total time UOWs shunted for commit/backout failure	The total time UOWs were shunted for commit/backout failure. Source field: RMGTSHTR
Current UOWs shunted for commit/backout failure	The current number of UOWs shunted for commit/backout failure. Source field: RMGCSHRO
Total time current UOWs shunted for commit/backout failure	The total time for the current UOWs shunted for commit/backout failure. Source field: RMGCSHTR
Indoubt Action Forced by Trandef	The number of forced indoubt action resolutions due to the transaction definition specifying that it cannot support indoubt waiting. Source field: RMGIAFTR
Indoubt Action Forced by Timeout	The number of forced indoubt action resolutions due to the indoubt wait timing out. Source field: RMGIAFTI
Indoubt Action Forced by No Wait	The number of forced indoubt action resolutions due to a recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGIAFNW

Table 269. Fields in the Recovery Manager report (continued)

Field Heading	Description
Indoubt Action Forced by Operator	The number of forced indoubt action resolutions due to the operator (CEMT or SPI command) cancelling the wait for indoubt resolution. Source field: RMGIAFOP
Indoubt Action Forced by Other	The number of forced indoubt action resolutions due to reasons other than those already referenced in this table. Source field: RMGIAFOT
The following fields are a breakdown of 'Indoubt Action Forced by No Wait':	
Indoubt Action Forced by TD Queues	The number of forced indoubt action resolutions due to a recoverable transient data queue being unable to support indoubt waiting. Source field: RMGNWTD
Indoubt Action Forced by LU61 Connections	The number of forced indoubt action resolutions due to the use of an LU6.1 intersystem link, which is unable to support indoubt waiting. Source field: RMGNW61
Indoubt Action Forced by MRO Connections	The number of forced indoubt action resolutions due to the use of an MRO connection, which is unable to support indoubt waiting. Source field: RMGNWMRO
Indoubt Action Forced by RMI Exits	The number of forced indoubt action resolutions due to an RMI exit being unable to support indoubt waiting. Source field: RMGNWRMI
Indoubt Action Forced by Other	The number of forced indoubt action resolutions due to another recoverable resource or resource manager coordinator being unable to support indoubt waiting. Source field: RMGNWOTH
Number of Indoubt Action Mismatches	The number of forced indoubt action resolutions that a participating resource manager coordinator resolved in the opposite way to CICS. Source field: RMGIAMIS

Storage reports

There are five storage reports. The storage reports provide information about the use of MVS and CICS virtual storage. There are separate reports for storage below 16 MB, storage above 16 MB but below 2 GB, and storage above 2 GB.

Storage below 16 MB report

The Storage below 16 MB report provides information on the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage below 16 MB and helps you to verify the size values used for the CDSA, UDSA, SDSA, and RDSA and the value set for the DSA limit.

Table 270. Fields in the Storage below 16 MB report

Field Heading	Description
Region size established from REGION= parameter	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	**Description**

Table 270. Fields in the Storage below 16 MB report (continued)

Field Heading	Description
Private Area Region size below 16MB	The private area size below 16 MB, expressed in KB.
Max LSQA/SWA storage allocated below 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the scheduler work area (SWA) subpools below 16 MB, expressed in KB.
Max User storage allocated below 16MB (VIRT)	The maximum amount of virtual storage allocated from the user subpools below 16 MB, expressed in KB.
System Use	An amount of virtual storage available for system use.
RTM	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	The amount of storage below 16 MB that could be allocated by increasing the DSALIM parameter or by MVS storage GETMAIN requests.
MVS PVT Size	The maximum MVS private area (PVT) size below 16 MB, expressed in KB.
MVS CSA Size / Allocated	The MVS common system area (CSA) size and the amount of the MVS CSA allocated below 16 MB, expressed in KB.
MVS SQA Size / Allocated	The MVS system queue area (SQA) size and the amount of the MVS SQA allocated below 16 MB, expressed in KB.
Current DSA Limit	The current DSA Limit, expressed in KB. Source field: (SMSDSALIMIT / 1024)
Current Allocation for DSAs	The current amount of storage allocated to the DSAs below 16 MB, expressed in KB. This value may be smaller or larger than the current DSA limit. Source field: (SMSDSATOTAL / 1024)
VIRT minus Current DSA Limit	The total amount of user storage allocated/used below 16 MB minus the current DSA limit. This indicates the amount of user storage that is allocated below 16 MB, and is not allocated to the DSA. Source field: ((VIRT - SMSDSALIMIT) / 1024)
Peak Allocation for DSAs	The peak amount of storage allocated to the DSAs below 16 MB, expressed in KB. This value may be smaller or larger than the current DSA limit. Source field: (SMSHWMDSATOTAL / 1024)
Current DSA Size	The current size of the CDSA, UDSA, SDSA, or RDSA, expressed in KB. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA, expressed in KB. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	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	The peak amount of storage used in this DSA, expressed in KB. Source field: (SMSHWMPS / 1024)
Peak DSA Size	The peak size of the CDSA, UDSA, SDSA, or the RDSA, expressed in KB. Source field: (SMSHWMDSASZ / 1024)

Table 270. Fields in the Storage below 16 MB report (continued)

Field Heading	Description
Cushion Size	The size of the cushion, expressed in KB. The cushion forms part of the CDSA, UDSA, SDSA, or the RDSA, and is the amount of storage below which CICS goes short on storage (SOS). Source field: (SMSCSIZE / 1024)
Free Storage (inc. Cushion)	The current amount of free storage in this DSA, expressed in KB. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this DSA, expressed in KB. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	The lowest amount of free storage in this DSA, expressed in KB. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed in bytes. Source field: (SMSLFA / 1024)
Largest Free Area as % of DSA	The largest contiguous free area in the CDSA, UDSA, SDSA, or RDSA, expressed as a percentage of the current DSA size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Free/Free Storage	An indication of the storage fragmentation in this DSA. This value is calculated by dividing the "Largest Free Area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is small, this DSA is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The number of extents currently allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to the DSA since the last time statistics were recorded. Source field: SMSEXTSA
Number of extents released	The number of extents that were released from the DSA since the last time statistics were recorded. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the CDSA, UDSA, SDSA, or RDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSASR

Table 270. Fields in the Storage below 16 MB report (continued)

Field Heading	Description
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the CDSA, UDSA, SDSA, or RDSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at that moment.. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests that are currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests that were suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests that were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	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 and there are no more free extents available to increase the size of this DSA. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this DSA, where SOS means that the cushion is currently in use, or at least one task is suspended for storage, or both. This field applies to the CDSA, UDSA, SDSA, and RDSA. Source field: SMSSOS
Total time Short-On-Storage	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the DSA. This field applies to the CDSA, UDSA, SDSA, and RDSA. Source field: SMSSV
Access	The type of access of the DSA. Values are CICS, USER, or READONLY. If storage protection is not active, storage areas revert to an access type of 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

Storage above 16 MB report

The Storage above 16 MB report provides information about the use of MVS and CICS virtual storage. It contains the information you need to understand your current use of virtual storage between 16 MB and 2 GB (31-bit storage, also known as storage above the line). This report helps you to verify the size values used for the ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA, and the value set for the EDSA limit.

This report is produced using the **EXEC CICS COLLECT STATISTICS STORAGE** command. The statistics data is mapped by the DFHSMSDS DSECT.

Table 271. Fields in the Storage above 16 MB report

Field Heading	Description
Private Area Region size above 16MB	The private area size above 16 MB, expressed in KB.
Max LSQA/SWA storage allocated above 16MB (SYS)	The maximum amount of virtual storage allocated from the local system queue area (LSQA) and the SWA subpools above 16 MB, expressed in KB.
Max User storage allocated above 16MB (EXT)	The maximum amount of virtual storage allocated from the user subpools above 16 MB, expressed in KB.
Private Area Storage available above 16MB	The amount of storage above 16 MB that could be allocated by increasing the EDSALIM parameter or by MVS storage GETMAIN requests.
EXT minus Current EDSA Limit	The total amount of user storage allocated or used above 16 MB minus the current EDSA limit. This value indicates the amount of user storage that is allocated above 16 MB, but is not allocated to the EDSA. Source field: ((EXT - SMSEDSALIMIT) / 1024)
MVS EPVT size	The maximum extended MVS private area (EPVT) size above 16 MB, expressed in KB.
MVS ECSA Size / Allocated	The MVS extended common service area (ECSA) size and the amount of the MVS CSA allocated above 16 MB, expressed in KB.
MVS ESQA Size / Allocated	The MVS extended system queue (ESQA) size and the amount of the MVS SQA allocated above 16 MB, expressed in KB.
Requests for MVS storage causing waits	The total number of MVS storage requests that have waited for MVS storage above 16 MB. Source field: SMSMVSSTGREQWAITS
Total time waiting for MVS storage	The total time that MVS storage requests have spent waiting for MVS storage above 16 MB. Source field: SMSTIMEWAITMVS
Current EDSA Limit	The current limit of the CICS extended dynamic storage areas, as defined by the **EDSALIM** system initialization parameter. This value is expressed in KB. Source field: (SMSEDSALIMIT / 1024)
Current Allocation for EDSAs	The total amount of storage currently allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than “Current EDSA limit”. This value is expressed in KB and might be smaller or larger than the current EDSA limit. Source field: (SMSEDSATOTAL / 1024)

Table 271. Fields in the Storage above 16 MB report (continued)

Field Heading	Description
Peak Allocation for EDSAs	The peak amount of storage allocated to the DSAs above 16 MB but below 2 GB (above the line). This value might be smaller or larger than "Current EDSA limit". This value is expressed in KB and might be smaller or larger than the current EDSA limit. Source field: (SMSHWMEDSATOTAL / 1024)
Current DSA Size	The current size of the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA, expressed in KB. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA, expressed in KB. Source field: ((SMSDSASZ - SMSFSTG) / 1024)
Current DSA Used as % of DSA	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	The peak amount of storage used in this DSA, expressed in KB. Source field: (SMSHWMPS / 1024)
Peak DSA Size	The peak size of the ECDSA, EUDSA, ESDSA, ETDSA or the ETDSA, expressed in KB. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	The size of the cushion, expressed in KB. The cushion forms part of the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA and is the amount of storage below which CICS goes SOS. Source field: (SMSCSIZE / 1024)
Free Storage (inc. Cushion)	The current amount of free storage in this DSA, expressed in KB. Source field: (SMSFSTG / 1024)
Peak Free Storage	The peak amount of free storage in this DSA, expressed in KB. Source field: (SMSHWMFSTG / 1024)
Lowest Free Storage	The lowest amount of free storage in this DSA, expressed in KB. Source field: (SMSLWMFSTG / 1024)
Largest Free Area	The length of the largest contiguous free area in the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA, expressed in KB. Source field: (SMSLFA / 1024)
Largest Free Area as % of DSA	The largest contiguous free area in the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA, expressed as a percentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Free/Free Storage	An indication of the storage fragmentation in this DSA. This value is calculated by dividing the "Largest free area" (SMSLFA) by the "Free storage" (SMSFSTG). If the ratio is small, this DSA is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The number of extents currently allocated to this DSA. Source field: SMSEXTS

Table 271. Fields in the Storage above 16 MB report (continued)

Field Heading	Description
Number of extents added	The number of extents added to the DSA since the last time statistics were recorded. Source field: SMSEXTSA
Number of extents released	The number of extents that were released from the DSA since the last time statistics were recorded. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA. Source field: SMSCSUBP
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA. Source field: SMSASR
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from the ECDSA, EUDSA, ESDSA, ERDSA, or ETDSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at that moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests that are currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests that were suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests that were purged while suspended for storage. Source field: SMSPWWS
Times cushion released	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 and there are no more free extents available to increase the size of this DSA. Source field: SMSCREL

Table 271. Fields in the Storage above 16 MB report (continued)

Field Heading	Description
Times Short-On-Storage	The number of times CICS went SOS in this DSA, where SOS means that the cushion is currently in use, or at least one task is suspended for storage, or both. This field applies to the ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA. Source field: SMSSOS
Total time Short-On-Storage	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage Violations	The number of storage violations recorded in the DSA. This field applies to the ECDSA, EUDSA, ESDSA, ERDSA, and ETDSA. Source field: SMSSV
Access	The type of access of the DSA. Values are CICS, USER, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the ERDSA. • CICS - access is CICS key • USER - access is USER key • READONLY - access is read-only protection • TRUSTED - access is CICS key. Source field: SMSACCESS

Storage above 2 GB report

The Storage above 2 GB report provides information about the use of MVS and CICS virtual storage. It contains the information you require to understand the use of 64-bit virtual storage, also known as storage above the bar. This report helps you to verify the allocation of storage for the CICS dynamic storage areas above the bar (GDSA) and for the CICS functions that use 64-bit storage.

This report is produced using the **EXEC CICS COLLECT STATISTICS STORAGE** command. The statistics data is mapped by the DFHSMSDS DSECT.

Table 272. Fields in the Storage above 2 GB report (part 1)

Field Heading	Description
MEMLIMIT Size	The value of the z/OS **MEMLIMIT** parameter, which limits the amount of 64-bit storage for the CICS region. This value can be in megabytes, gigabytes, terabytes, petabytes, or exabytes, depending on size. A value of `NOLIMIT` indicates that no upper limit is imposed. Source field: SMSMEMLIMIT
MEMLIMIT Set By	The source of the **MEMLIMIT** value: SMFPRM indicates that **MEMLIMIT** is set by `SYS1.PARMLIB(SMFPRMxx)`. JCL indicates that **MEMLIMIT** is set by JCL. REGION indicates that **MEMLIMIT** is set to NOLIMIT because REGION=0M is specified in JCL. IEFUSI indicates that **MEMLIMIT** is set by the z/OS installation exit IEFUSI. Source field: SMSMEMLIMITSRC

Table 272. Fields in the Storage above 2 GB report (part 1) (continued)

Field Heading	Description
Current Address Space active (bytes)	The current address space available above the bar. This value is expressed in bytes. Source field: (SMSASACTIVE x 1048576)
Current Address Space active	The current address space available above the bar. This value is expressed in megabytes. Source field: SMSASACTIVE
Peak Address Space active	The peak amount of address space available above the bar. This value is expressed in megabytes. Source field: SMSHWMASACTIVE
MEMLIMIT minus Current Address Space active	The value of the z/OS **MEMLIMIT** parameter minus the current address space available above the bar. This value is expressed in megabytes. Source field: (SMSMEMLIMIT - SMSASACTIVE)
MEMLIMIT minus allocated to Private Memory Objects	The value of the z/OS **MEMLIMIT** parameter minus the amount of storage allocated from large virtual memory in private memory objects. This value is expressed in megabytes. Source field: (SMSMEMLIMIT - (SMSLVABYTES / 1048576))
Number of Private Memory Objects	The number of private memory objects allocated.[1] Source field: SMSLVNMEMOBJ
....minus Current GDSA extents	The number of private memory objects allocated minus the number of extents currently allocated to this DSA. Source field: (SMSLVNMEMOBJ - SMSEXTS)
Bytes allocated to Private Memory Objects	The number of bytes allocated from large virtual memory in private memory objects. This value is expressed in megabytes and bytes.[1] Source field: SMSLVABYTES
....minus Current GDSA allocated	The number of bytes allocated from large virtual memory in private memory objects minus the total storage currently allocated to the DSAs above the bar. This value is expressed in megabytes and bytes. Source field: (SMSLVABYTES - SMSGDSAALLOC)
Bytes hidden within Private Memory Objects	The number of bytes hidden in large virtual memory private memory objects. This value is expressed in megabytes and bytes.[1] Source field: SMSLVHBYTES
....minus Current GDSA hidden	The number of bytes hidden in large virtual memory private memory objects minus the storage allocated to the DSAs above the bar that is not currently active. This value is expressed in megabytes and bytes. Source field: (SMSLVHBYTES - (SMSGDSAALLOC - SMSGDSAACTIVE))
....minus CICS Internal Trace Table hidden	The number of bytes hidden in large virtual memory private memory objects minus the storage allocated to the DSAs above the bar that is not currently active, and minus the size of the CICS internal trace table. This value is expressed in megabytes. Source field: (SMSLVHBYTES - ((SMSGDSAALLOC - SMSGDSAACTIVE) + EXEC CICS INQUIRE TRACEDEST TABLESIZE)) / 1048756

Table 272. Fields in the Storage above 2 GB report (part 1) (continued)

Field Heading	Description
Bytes usable within Private Memory Objects	The number of usable bytes in large virtual memory private memory objects, that is, the number of bytes allocated minus the number of bytes hidden in large virtual memory private memory objects. This value is expressed in megabytes and bytes. Source field: (SMSLVABYTES - SMSLVHBYTES)
Peak bytes usable within Private Memory Objects	The high-water mark of usable bytes in large virtual memory private memory objects. This value is expressed in megabytes and bytes.[1] Source field: SMSLVGBYTES
Current GDSA Allocated	The total amount of storage currently allocated to the DSAs above the bar. This value is expressed in megabytes and bytes. Source field: SMSGDSAALLOC
Peak GDSA Allocated	The peak amount of storage allocated to the DSAs above the bar. This value is expressed in megabytes. Source field: SMSHWMGDSAALLOC
Current GDSA Active	The current storage in use above the bar. This value is expressed in megabytes and bytes. Source field: SMSGDSAACTIVE
Peak GDSA Active	The peak amount of storage in use above the bar. This value is expressed in megabytes. Source field: SMSHWMGDSAACTIVE
Current GDSA Used	The current amount of storage used in this DSA, expressed in megabytes. Source field: (SMSDSASZ - SMSFSTG)
Number of Shared Memory Objects	The number of shared memory objects allocated.[1] Source field: SMSLVSHRNMEMOBJ
Bytes allocated to Shared Memory Objects	The number of shared bytes allocated from high virtual memory. This value is expressed in megabytes and bytes.[1] Source field: SMSLVSHRBYTES
Peak bytes usable within Shared Memory Objects	The high-water mark for the number of shared bytes in large virtual memory objects. This value is expressed in megabytes and bytes.[1] Source field: SMSLVSHRGBYTES
Auxiliary Slots backing Private Memory Objects	The number of auxiliary storage slots that are used to back 64-bit private memory objects.[1] Source field: SMSHVAUXSLOTS
HWM Auxiliary Slots backing Private Memory Object	The high-water mark of auxiliary storage slots that are used to back 64-bit private memory objects.[1] Source field: SMSHVGAUXSLOTS
Real Frames backing Private Memory Objects	The number of real storage frames that are used to back 64-bit private memory objects.[1] Source field: SMSHVPAGESINREAL

Table 272. Fields in the Storage above 2 GB report (part 1) (continued)

Field Heading	Description
HWM Real Frames backing Private Memory Objects	The high-water mark for the number of real storage frames that are used to back 64-bit private memory objects.[1] Source field: SMSHVGPAGESINREAL
Number of Large Memory Objects Allocated	The number of large memory objects allocated by this address space.[1] Source field: SMSLARGEMEMOBJ
Number of Large Pages backed in Real Storage	The number of large pages (1 MB pages) backed in real storage owned by this address space.[1] Source field: SMSLARGEPAGESINREAL
CICS Internal Trace table size (bytes)	The current size set for the CICS internal trace table, expressed in bytes. Source field: **EXEC CICS INQUIRE TRACEDEST TABLESIZE**
CICS Internal Trace table size	The current size set for the CICS internal trace table, expressed in KB. Source field: **(EXEC CICS INQUIRE TRACEDEST TABLESIZE) / 1024**
IARV64 GETSTOR request size	The GETSTOR request size. This value is expressed in megabytes. Source field: SMSGETSTORSIZE
Number of IARV64 FROMGUARD failures	The number of times that a request for 64-bit storage has failed, where the request uses the z/OS IARV64 macro with the REQUEST=CHANGEGUARD, CONVERT=FROMGUARD parameters. Source field: SMSFROMGUARDFAIL
Largest IARV64 FROMGUARD failure size	The size of the largest request for 64-bit storage that has failed, in bytes, where the request uses the z/OS IARV64 macro with the REQUEST=CHANGEGUARD, CONVERT=FROMGUARD parameters. Source field: SMSFROMGUARDFAILSIZE

Table 273. Fields in the Storage above 2 GB report (part 2)

Field Heading	Description
MEMLIMIT Size	The value of the z/OS **MEMLIMIT** parameter, which limits the amount of 64-bit storage for the CICS region. This value can be in megabytes, gigabytes, terabytes, petabytes, or exabytes, depending on size. A value of NOLIMIT indicates that no upper limit is imposed. Source field: SMSMEMLIMIT
MEMLIMIT Set By	The source of the **MEMLIMIT** value: SMFPRM indicates that **MEMLIMIT** is set by SYS1.PARMLIB(SMFPRMxx). JCL indicates that **MEMLIMIT** is set by JCL. REGION indicates that **MEMLIMIT** is set to NOLIMIT because REGION=0M is specified in JCL. IEFUSI indicates that **MEMLIMIT** is set by the z/OS installation exit IEFUSI. Source field: SMSMEMLIMITSRC
Current Address Space active	The current address space available above the bar. This value is expressed in megabytes. Source field: SMSASACTIVE

Table 273. Fields in the Storage above 2 GB report (part 2) (continued)

Field Heading	Description
Peak Address Space active	The peak amount of address space available above the bar. This value is expressed in megabytes. Source field: SMSHWMASACTIVE
Current GDSA Allocated	The total amount of storage currently allocated to the DSAs above the bar. This value is expressed in megabytes. Source field: SMSGDSAALLOC
Peak GDSA Allocated	The peak amount of storage allocated to the DSAs above the bar. This value is expressed in megabytes. Source field: SMSHWMGDSAALLOC
Current GDSA Active	The current storage in use above the bar. This value is expressed in megabytes. Source field: SMSGDSAACTIVE
Peak GDSA Active	The peak amount of storage in use above the bar. This value is expressed in megabytes. Source field: SMSHWMGDSAACTIVE
CICS Internal Trace table size (bytes)	The current size set for the CICS internal trace table, expressed in bytes. Source field: `EXEC CICS INQUIRE TRACEDEST TABLESIZE`
CICS Internal Trace table size	The current size set for the CICS internal trace table, expressed in KB. Source field: `(EXEC CICS INQUIRE TRACEDEST TABLESIZE) / 1024`
Number of Private Memory Objects	The number of private memory objects allocated.[1] Source field: SMSLVNMEMOBJ
Bytes allocated to Private Memory Objects	The number of bytes allocated from large virtual memory in private memory objects.[1] Source field: SMSLVABYTES
Bytes hidden within Private Memory Objects	The number of bytes hidden in large virtual memory private memory objects.[1] Source field: SMSLVHBYTES
Peak bytes usable within Private Memory Objects	The high-water mark of usable bytes in large virtual memory private memory objects.[1] Source field: SMSLVGBYTES
Current DSA Size	The current size of the GCDSA, GUDSA, or GSDSA, expressed in megabytes. Source field: (SMSDSASZ / 1024)
Current DSA Used	The current amount of storage used in this DSA, expressed in megabytes. Source field: (SMSDSASZ - SMSFSTG)
Current DSA Used as % of DSA	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	The peak amount of storage used in this DSA, expressed in megabytes. Source field: SMSHWMPS

Table 273. Fields in the Storage above 2 GB report (part 2) (continued)

Field Heading	Description
Peak DSA Size	The peak size of this DSA, expressed in megabytes. Source field: (SMSHWMDSASZ / 1024)
Cushion Size	The size of the cushion for this DSA, expressed in megabytes. The cushion forms part of each DSA and is the amount of storage below which CICS goes SOS. Source field: SMSCSIZE
Free Storage (inc. Cushion)	The amount of free storage in this DSA; that is, the number of free pages multiplied by the page size (1 MB), expressed in megabytes. Source field: SMSFSTG
Peak Free Storage	The largest amount of storage that is free in this DSA since the last time that statistics were recorded, expressed in megabytes. Source field: SMSHWMFSTG
Lowest Free Storage	The smallest amount of storage that is free in this DSA since the last time that statistics were recorded, expressed in megabytes. Source field: SMSLWMFSTG
Largest Free Area	The length of the largest contiguous free area in this DSA, expressed in megabytes. Source field: SMSLFA
Largest Free Area as % of DSA	The largest contiguous free area in this DSA, expressed as a percentage of the current DSA Size. Source field: ((SMSLFA / SMSDSASZ) * 100)
Largest Free/Free Storage	An indication of the storage fragmentation in this DSA. This value is calculated by dividing the Largest free area (SMSLFA) by the Free storage (SMSFSTG). If the ratio is small, this DSA is fragmented. Source field: (SMSLFA / SMSFSTG)
Current number of extents	The number of extents currently allocated to this DSA. Source field: SMSEXTS
Number of extents added	The number of extents added to the DSA since the last time statistics were recorded. Source field: SMSEXTSA
Number of extents released	The number of extents that were released from the DSA since the last time statistics were recorded. Source field: SMSEXTSR
Getmain Requests	The number of GETMAIN requests from this DSA. Source field: SMSGMREQ
Freemain Requests	The number of FREEMAIN requests from this DSA. Source field: SMSFMREQ
Current number of Subpools	The current number of subpools (domain and task) in this DSA. Source field: SMSCSUBP

Table 273. Fields in the Storage above 2 GB report (part 2) (continued)

Field Heading	Description
Add Subpool Requests	The number of ADD_SUBPOOL requests to create a subpool (domain or task) from this DSA. Source field: SMSASR
Delete Subpool Requests	The number of DELETE_SUBPOOL requests (domain or task) from this DSA. Source field: SMSDSR
Times no storage returned	The number of times a GETMAIN request with SUSPEND(NO) returned the condition INSUFFICIENT_STORAGE. Source field: SMSCRISS
Times request suspended	The number of times a GETMAIN request with SUSPEND(YES) was suspended because of insufficient storage to satisfy the request at that moment. Source field: SMSUCSS
Current requests suspended	The number of GETMAIN requests that are currently suspended for storage. Source field: SMSCSS
Peak requests suspended	The peak number of GETMAIN requests that were suspended for storage. Source field: SMSHWMSS
Requests purged while waiting	The number of requests that were purged while suspended for storage. Source field: SMSPWWS
Times Cushion released	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 and there are no more free extents available to increase the size of this DSA. Source field: SMSCREL
Times Short-On-Storage	The number of times CICS went SOS in this DSA, where SOS means that the cushion is currently in use, or at least one task is suspended for storage, or both. Source field: SMSSOS
Total time Short-On-Storage	The accumulated time that CICS has been SOS in this DSA. Source field: SMSTSOS
Average Short-On-Storage time	The average time that CICS has been SOS in this DSA. Source field: (SMSTSOS / SMSSOS)
Storage violations	The number of storage violations recorded in the DSA. Source field: SMSSV
Access	The type of access of this DSA. Values are CICS or USER. If storage protection is not active, storage areas revert to an access type of CICS. • CICS - access is CICS key • USER - access is user key Source field: SMSACCESS

Note:

1. For more information about the memory that this statistic refers to, see Using the 64-bit Address Space in the z/OS MVS Programming: Extended Addressability Guide.

Storage - Domain Subpools reports

The storage subpool reports provide statistics about CICS domain and task storage subpool allocations and use.

The subpool reports are in two parts:

- Domain subpools, consisting of only those storage domain subpools that are allocated in the CICS, read-only, and shared dynamic storage areas (DSAs), that is, the CDSA, RDSA, SDSA, ECDSA, ERDSA, ESDSA, ETDSA, GCDSA, and GSDSA. The information for this report is collected by using the **EXEC CICS INQUIRE SUBPOOL** and **EXEC CICS COLLECT STATISTICS SUBPOOL** commands. The domain subpools are split into two reports, with some shared fields, to represent all domain subpools information.
- Task subpools, consisting of only those subpools allocated for user task lifetime storage. The information for this report is collected using the **EXEC CICS COLLECT STATISTICS TASKSUBPOOL** command.

Table 274. Fields in the Storage - Domain Subpools report (Part 1)

Field Heading	Description
Subpool Name	The unique 8-character name of the domain subpool. The values of the domain subpool field are described in CICS subpools in Improving performance. Source field: SMDSPN
Location	The name of the DSA that the domain subpool is allocated from. Values can be CDSA, SDSA, RDSA, ECDSA, ESDSA, ERDSA, ETDSA, GCDSA, or GSDSA. Source field: SMDDSANAME
Access	The type of access of the subpool. Values are CICS, READONLY, or TRUSTED. If storage protection is not active, storage areas revert to an access type of CICS, except for those in the RDSA or ERDSA. • SMDCICS (X'01') access is CICS key. • SMDREADONLY (X'03') is read-only protection. • SMDTRUSTED (X'04') access is CICS key. Source field: SMDACCESS
Element Type	Indicates whether all elements in the subpool are fixed length or variable length. Source field: SMDETYPE
Element Length	The length of each subpool element (applicable to fixed length subpools only). For further information about subpool elements, see CICS subpools in Improving performance. Source field: SMDFLEN
Initial Free	The total number of kilobytes of the elements that are initially allocated when the domain subpool is preallocated. Source field: SMDIFREE

Table 274. Fields in the Storage - Domain Subpools report (Part 1) (continued)

Field Heading	Description
Current Elements	The current number of storage elements in the subpool. The number of elements remaining after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests. Source field: SMDCELEM
Current Element Stg	The sum of the lengths of all the elements in the subpool, expressed in bytes. Source field: SMDCES
Current Page Stg	The space taken by all the pages allocated to the subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Source field: SMDCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. This field does not apply to the GCDSA or GSDSA. Source field: ((SMDCPS / *dsasize*) * 100)
Peak Page Stg	The peak page storage allocated to support the storage requirements of this subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Source field: SMDHWMPS

Table 275. Fields in the Storage - Domain Subpools report (Part 2)

Field Heading	Description
Subpool Name	The unique 8-character name of the domain subpool. The values of the domain subpool field are described in CICS subpools in Improving performance. Source field: SMDSPN
Location	The name of the DSA that the domain subpool is allocated from. Values can be CDSA, SDSA, RDSA, ECDSA, ESDSA, ERDSA, ETDSA, GCDSA, or GSDSA. Source field: SMDDSANAME
Getmain Requests	The total number of GETMAIN requests for the subpool. Source field: SMDGMREQ
Freemain Requests	The total number of FREEMAIN requests for the subpool. Source field: SMDFMREQ
Current Element Stg	The sum of the lengths of all the elements in the subpool, expressed in bytes. Source field: SMDCES
Current Page Stg	The space taken by all the pages allocated to the subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Source field: SMDCPS
Peak Page Stg	The peak page storage allocated to support the storage requirements of this subpool, expressed in bytes (or megabytes for 64-bit (above-the-bar) storage). Source field: SMDHWMPS

Table 276. Fields in the Storage - Domain Subpool Totals report

Field Heading	Description
DSA Name	The abbreviated name of the CICS dynamic storage area to which the subpool totals apply. Source field: SMDSANAME
Number of Subpools	The total number of subpools in this DSA.
Getmain Requests	The total number of GETMAIN requests for subpools in this DSA. Source field: Total of SMDGMREQ values for each DSA.
Freemain Requests	The total number of FREEMAIN requests for subpools in this DSA. Source field: Total of SMDFMREQ values for each DSA.
Current Elements	The total number of elements remaining after FREEMAIN requests; that is, the difference between the total number of GETMAIN and FREEMAIN requests. Source field: Total of all SMDCELEM values for each DSA
Current Element Stg	The total amount of storage of the current elements, expressed in bytes. Source field: Total of all SMDCES values for each DSA.
Current Page Stg	The total amount of subpool page storage for all DSAs, expressed in kilobytes (or megabytes for 64-bit (above-the-bar) storage). Source field: Total of all SMDCPS values for each DSA.
% of DSA	The current element storage of all the subpools as a percentage of the DSA in which they reside. This field does not apply to the GCDSA or GSDSA. Source: ((Total of all SMDCPS values / *dsasize*) * 100)
% of DSA Limit	The current element storage of all the subpools as a percentage of the limit of DSA in which they reside. This field does not apply to the GCDSA or GSDSA. Source: ((Total of all SMDCPS values / *dsalimit*) * 100)

Table 277. Fields in the Task Subpools report

Field Heading	Description
Subpool Name	The name of the DSA page pool that contains the task storage. Source field: SMDSPN
Access	The type of access of the subpool. Access type can be CICS (key 8) or USER (key 9). Source field: SMTACCESS
Getmain Requests	The total number of task subpool GETMAIN requests from this dynamic storage area. That is, the number of GETMAIN requests issued for this subpool. Source field: SMTGMREQ
Freemain Requests	The total number of task subpool FREEMAIN requests from this dynamic storage area.. That is, the number of FREEMAIN requests issued for this subpool. Source field: SMTFMREQ

Table 277. Fields in the Task Subpools report (continued)

Field Heading	Description
Current Elements	The number of elements in all the task subpools in this dynamic storage area. That is, the number of elements remaining after FREEMAIN requests (the difference between the number of GETMAIN and FREEMAIN requests). Source field: SMTCNE
Current Element Stg	The sum of the storage occupied by all elements in task subpools in this dynamic storage area, expressed in bytes. Source field: SMTCES
Average Element Size	The average size in bytes of an element. Source field: (SMTCES / SMTCNE)
Current Page Stg	The sum of the storage in all pages allocated to task subpools in this dynamic storage area. This value is expressed in kilobytes. Source field: SMTCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. Source field: ((SMTCPS / *dsasize*) * 100)
Peak Page Stg	The peak page storage allocated to support task storage activity in this dynamic storage area. This value is expressed in kilobytes. Source field: SMTHWMPS

Storage - Program Subpools report

The Storage Subpools Report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command. The statistics data is mapped by the **DFHSMDDS DSECT**.

Table 278. Fields in the Storage - Program Subpools report

Field Heading	Description
Subpool Name	The name of the domain subpool. Source field: SMDSPN
Subpool Location	The DSA location of the domain subpool. Source field: SMDLOCN
Current Storage	The current amount of storage allocated to this domain subpool. Source field: SMDCPS
Peak Storage	The peak amount of storage allocated to this domain subpool. Source field: SMDHWMPS

System Status report

The System Status report is produced from various sources. The commands used are detailed in the table.

Table 279. Fields in the System Status report

Field Heading	Description
System Status	
MVS Product Name	The product level of MVS. Source field: MVS field CVTPRODN
CICS Transaction Server Level	The product version, release, and modification number of CICS Transaction Server. Source field: **EXEC CICS INQUIRE SYSTEM CICSTSLEVEL**
CICS Startup	The type of CICS startup. Source field:**EXEC CICS INQUIRE SYSTEM STARTUP**(*cvda*)**COLDSTATUS**(*cvda*)
MVS Workload Manager (WLM) Mode	The z/OS Workload Manager (WLM) mode that is in operation in the CICS region. Source field: MNG-WLM-MODE
CICS Status	The status of the local CICS system. Source field: **EXEC CICS INQUIRE SYSTEM CICSSTATUS**(*cvda*)
WLM Server	Indicates whether the CICS region is an z/OS Workload Manager server. Source field: MNG-SERVER-STATUS
CEC Machine Type and Model	The CEC machine type and model number for the physical hardware environment where the CICS region is running. Source field: MNG-CEC-MACHINE-TYPE, MNG-CEC-MODEL-NUMBER
WLM Manage Region Using Goals of	Whether z/OS Workload Manager manages the CICS address space using region goals, transaction goals, or both. Source field: MNG-WLM-AS-GOAL-MANAGEMENT
WLM Workload Name	The name of the workload defined for the CICS region. Source field: MNG-WORKLOAD-NAME
VTAM Open Status	The status of the z/OS Communications Server connection for this CICS system (VTAM is the previous name for z/OS Communications Server). Source field:**EXEC CICS INQUIRE VTAM OPENSTATUS**(*cvda*)
WLM Service Class	The class name of the z/OS Workload Manager service class for the CICS region. Source field: MNG-SERVICE-CLASS
IRC Status	The status of IRC for this CICS system. Source field: **EXEC CICS INQUIRE IRC OPENSTATUS**(*cvda*)
WLM Report Class	The name of the z/OS Workload Manager report class, if any. Source field: MNG-REPORT-CLASS
IRC XCF Group Name	The name of the cross-system coupling facility (XCF) group of which this region is a member. Source field: **EXEC CICS INQUIRE IRC XCFGROUP**(*data-area*)

Table 279. Fields in the System Status report (continued)

Field Heading	Description
WLM Resource Group	The name of the z/OS Workload Manager resource group, if any. Source field: MNG-RESOURCE-GROUP
WLM Goal Type	The z/OS Workload Manager goal type for the CICS address space. Source field: MNG-WLM-AS-GOAL-TYPE
Storage Protection	The status of storage protection. Source field: **EXEC CICS INQUIRE SYSTEM STOREPROTECT**(*cvda*)
WLM Goal Value	For an z/OS Workload Manager goal type of velocity, the goal value for the CICS address space. Source field: MNG-WLM-AS-GOAL-VALUE
Transaction Isolation	Indicates the status of transaction isolation. Source field: SMSTRANISO
WLM Goal Importance	The importance level of the z/OS Workload Manager goal for the CICS address space. 5 is lowest, 1 is highest. Source field: MNG-WLM-AS-GOAL-IMPORTANCE
Reentrant Programs	Whether read-only programs reside in key-0 protected storage. Source field: SMSRENTPGM
WLM CPU Critical	Whether long-term processor protection is assigned to the CICS address space in the z/OS Workload Manager. Source field: MNG-WLM-AS-CPU-CRITICAL
Exec storage command checking	Whether CICS validates start addresses of storage referenced as output parameters on EXEC CICS commands. Source field: **EXEC CICS INQUIRE SYSTEM CMDPROTECT**(*cvda*)
WLM Storage Critical	Whether long-term storage protection is assigned to the CICS address space in the z/OS Workload Manager. Source field: MNG-WLM-AS-STG-CRITICAL
Force Quasi-Reentrant	Whether CICS forces all user application programs specified as CONCURRENCY(THREADSAFE) to run under the CICS QR TCB. Source field:**EXEC CICS INQUIRE SYSTEM FORCEQR**(*cvda*)
RLS Status	The status of VSAM RLS for this CICS system. Source field: **EXEC CICS INQUIRE SYSTEM RLSSTATUS**(*cvda*)
RRMS/MVS Status	The status of RRMS/MVS for this CICS system. Source field: **EXEC CICS INQUIRE RRMS OPENSTATUS**(*cvda*)
Program Autoinstall	The status of program autoinstall. Source field: **EXEC CICS INQUIRE SYSTEM PROGAUTOINST**(*cvda*)
Terminal Autoinstall	The status of terminal autoinstall. Source field**EXEC CICS INQUIRE AUTOINSTALL**(*cvda*)
TCP/IP Status	The status of TCP/IP for this CICS system. Source field: **EXEC CICS INQUIRE TCPIP OPENSTATUS**(*cvda*)

Table 279. Fields in the System Status report (continued)

Field Heading	Description
Activity Keypoint Frequency	The current activity keypoint trigger value, which is the number of logging operations between the taking of keypoints. Source field: **EXEC CICS INQUIRE SYSTEM AKP**(*data area*).
Max IP Sockets	The maximum number of IP sockets that can be managed by the CICS sockets domain. Source field:**EXEC CICS INQUIRE TCPIP MAXSOCKETS()**
Logstream Deferred Force Interval	The current logstream deferred force interval. Source field: **EXEC CICS INQUIRE SYSTEM LOGDEFER()**
Active IP Sockets	The current number of IP sockets managed by the CICS sockets domain. Source field: **EXEC CICS INQUIRE TCPIP ACTSOCKETS()**
DB2 Connection Name	The name of the currently installed DB2 connection. Source field: **EXEC CICS INQUIRE SYSTEM DB2CONN**(*data area*)
DB2 Connection Status	The status of the CICS-DB2 connection. Source field: **EXEC CICS INQUIRE DB2CONN() CONNECTST**(*cvda*)
WEB Garbage Collection Interval	The current interval at which the web garbage collection task runs to clean up web 3270 state data. Source field: **EXEC CICS INQUIRE WEB GARBAGEINT()**
Terminal Input timeout Interval	The current period after which inactive web 3270 sessions are eligible for garbage collection. Source field: **EXEC CICS INQUIRE WEB TIMEOUTINT()**
Monitoring	
Monitoring	Whether CICS monitoring is active in the system. Source field: **EXEC CICS INQUIRE MONITOR STATUS**(*cvda*)
Exception Class	Whether the exception class of CICS monitoring data is being collected. Source field: **EXEC CICS INQUIRE MONITOR EXCEPTCLASS**(*cvda*)
Performance Class	Whether the performance class of CICS monitoring data is being collected. Source field: **EXEC CICS INQUIRE MONITOR PERFCLASS**(*cvda*)
Resource Class	Whether the transaction resource class of CICS monitoring data is being collected. Source field: **EXEC CICS INQUIRE MONITOR RESRCECLASS**(*cvda*)
Identity Class	Whether the identity class of CICS monitoring data is being collected. Source field: **EXEC CICS INQUIRE MONITOR IDNTYCLASS**(*cvda*)
Data Compression Option	Whether data compression is active for the SMF 110 monitoring records output by CICS. Source field: MNG-COMPRESSION-OPTION
Application Naming	Whether CICS application support is enabled. Source field: **EXEC CICS INQUIRE MONITOR APPLNAMEST**(*cvda*)

Table 279. Fields in the System Status report (continued)

Field Heading	Description
RMI Option	Whether performance monitoring data is being collected for the resource managers used by your transaction. Source field: **EXEC CICS INQUIRE MONITOR RMIST**(*cvda*)
Converse Option	Whether a performance class record is being written each time a conversational task waits for terminal input as well as at task end, or if a single performance class record is being written for the combined terminal waits. Source field:**EXEC CICS INQUIRE MONITOR CONVERSEST**(*cvda*)
Syncpoint Option	Whether performance monitoring data is being recorded separately for each unit of work (UOW) in tasks that contain multiple UOWs, or if performance monitoring data is being combined over all UOWs in a single task for recording. Source field:**EXEC CICS INQUIRE MONITOR SYNCPOINTST**(*cvda*)
Time Option	Whether the performance class time-stamp fields returned to an application using the COLLECT STATISTICS MONITOR command are expressed in local or GMT. Source field: **EXEC CICS INQUIRE MONITOR TIME**(*cvda*)
DPL Resource Limit	The maximum number of distributed program links for which transaction resource monitoring is being performed. Source field: **EXEC CICS INQUIRE MONITOR DPLLIMIT**(*cvda*)
File Resource Limit	The maximum number of files for which transaction resource monitoring is being performed. Source field: **EXEC CICS INQUIRE MONITOR FILELIMIT**(*cvda*)
Tsqueue Resource Limit	The maximum number of temporary storage queues for which transaction resource monitoring is being performed. Source field:**EXEC CICS INQUIRE MONITOR TSQUEUELIMIT**(*cvda*)
Exception Class Records	The number of exception records written to SMF. Source field: MNGER
Exception Class Suppressed	The number of exception records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGERS
Performance Class Records	The number of performance records scheduled for output to SMF. The monitoring domain buffers performance class records. If monitoring is deactivated, the performance class records that have been buffered are not in the report. Source field: MNGPR
Performance Records Suppressed	The number of performance records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGPRS
Resource Class Records	The number of transaction resource records scheduled for output to SMF. The monitoring domain buffers transaction resource class records. If monitoring is deactivated, the transaction resource class records that have been buffered are not in the report. Source field: MNGRR

Table 279. Fields in the System Status report (continued)

Field Heading	Description
Resource Records Suppressed	The number of transaction resource records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGRRS
Identity Class Records	The number of identity class records scheduled for output to SMF. The monitoring domain buffers identity class records. If monitoring is deactivated, the identity class records that have been buffered are not in the report. Source field: MNGIR
Identity Records Suppressed	The number of identity class records suppressed by a global user exit program at exit point XMNOUT. Source field: MNGIRS
Monitoring SMF Records	The number of monitoring 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 each SMF record has one exception record. The performance class, for example, has many performance class records per SMF record. The SMF record for the performance class is written when the buffer is full, performance class is deactivated, or CICS is quiescing. Source field: MNGSMFR
Monitoring SMF Errors	The number of non-OK responses from the request to write a monitoring record to SMF. This count is incremented when an SMF write fails for any reason; for example, when SMF is inactive. Source field: MNGSMFE
Monitoring SMF Records Compressed	The number of compressed monitoring records written to the SMF data set. This information is collected only when data compression for monitoring records is active. Source field: MNGSMFCM
Monitoring SMF Records Not Compressed	The number of monitoring records written to the SMF data set for which data compression was not performed. This information is collected only when data compression for monitoring records is active. Source field: MNGSMFNC
Percentage of SMF Records Compressed	The percentage of monitoring records written to the SMF data set which were compressed. This information is collected only when data compression for monitoring records is active. Source field: (MNGSMFCM / (MNGSMFCM + MNGSMFNC)) * 100
Statistics	
Statistics Recording	The status of statistics recording. Source field: `EXEC CICS INQUIRE STATISTICS RECORDING`(*cvda*)
Statistics Last Reset Time	The time of the last statistics reset. Source field: `EXEC CICS EXTRACT STATISTICS LASTRESET()`
Elapsed Time Since Reset	The elapsed time since the last statistics reset.
Statistics Interval	The current statistics recording interval. Source field: `EXEC CICS INQUIRE STATISTICS INTERVAL`

Table 279. Fields in the System Status report (continued)

Field Heading	Description
Next Statistics Collection	The next statistics recording time. Source field: **EXEC CICS INQUIRE STATISTICS NEXTTIME**
Statistics End-of-Day Time	The current end-of-day time for recording statistics. Source field: **EXEC CICS INQUIRE STATISTICS ENDOFDAY**
Statistics Start Date and Time	The current start date and time for recording statistics. Source field: STGCSTRT
Statistics SMF Records	The number of suppressed requests to write a statistics record to SMF. Source field: STGSMFS
Statistics SMF Writes Suppressed	The number of statistics SMF records written to the SMF data set. Source field: STGSMFW
Statistics SMF Errors	The number of non-OK responses from the request to write a statistics record to SMF. This count is incremented when an SMF write fails for any reason; for example, when SMF is inactive. Source field: STGSMFE
Current tasks at last attach	The current number of user transactions attached in the region at the time of the last transaction attached. Source field: MNGCAUTA
MXT value at last attach	The current MXT value at the time of the last transaction attached. Source field: MNGMXUTA
Time last user transaction attached	The date and time of the last user transaction attached. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been attached since the statistics were last reset. Source field: MNGLUTAT
Time last user transaction ended	The date and time that the last user transaction ended. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been attached since the statistics were last reset. Source field: MNGLUTCL
System transactions	The number of system transactions that have ended durng the statistics interval. Source field: MNGSTNUM
User transactions ended	The number of user transactions that have ended during the statistics interval. Source field: MNGUTNUM
Total transactions ended	The total number of system and user trasactions that have ended during the statistics interval. Source field: (MNGSTNUM + MNGUTNUM)
Average user transaction resp time	The rolling average user transaction response time. Source field: MNGAUTRT

Table 279. Fields in the System Status report (continued)

Field Heading	Description
Peak user transaction resp time	The maximum user transaction response time. Source field: MNGPUTRT
Peak user transaction resp time at	The date and time of the maximum user transaction response time. Source field: MNGLUTRT
Total transaction CPU time	The total transaction CPU time accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Source field: MNGCPUT
Total transaction CPU time on CP	The total transaction CPU time on a standard processor accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Source field: MNGTONCP
Total transaction CPU offload on CP	The total transaction CPU time on a standard processor but was eligible for offload to a specialty processor (zIIP or zAAP) accumulated for the CICS dispatcher managed TCB modes used by the transactions that have completed during the interval. Source field: MNGOFLCP
Average Compressed Record Length	The rolling average compressed record length for monitoring records written to the SMF data set, calculated from those monitoring records that were compressed. This information is collected only when data compression for monitoring records is active. Source field: MNGAVCRL
Average Uncompressed Record Length	The rolling average record length for monitoring records written to the SMF data set for which data compression was not performed. This information is collected only when data compression for monitoring records is active. Source field: MNGAVURL
Average Record Compression Percent	The average record length compression percentage. This information is collected only when data compression for monitoring records is active. Source field: (MNGAVURL - MNGAVCRL) / MNGAVURL) * 100
Trace Status	
Internal Trace Status	The status of internal tracing. Source field: **EXEC CICS INQUIRE TRACEDEST INTSTATUS**(*cvda*)
Auxiliary Trace Status	The status of auxiliary tracing. Source field: **EXEC CICS INQUIRE TRACEDEST AUXSTATUS**(*cvda*)
GTF Trace Status	The status of GTF tracing. Source field: **EXEC CICS INQUIRE TRACEDEST GTFSTATUS**(*cvda*)
Internal Trace Table Size	The current size of the internal trace table. Source field: **EXEC CICS INQUIRE TRACEDEST TABLESIZE**
Current Auxiliary Dataset	The name of the current auxiliary trace data set. Source field: **EXEC CICS INQUIRE TRACEDEST CURAUXDS**(*cvda*)

Table 279. Fields in the System Status report (continued)

Field Heading	Description
Auxiliary Switch Status	The status of the auxiliary trace autoswitch facility. Source field:**EXEC CICS INQUIRE TRACEDEST SWITCHSTATUS**(*cvda*)
Dumps	
System Dumps	The number of system dumps taken. Source field: SDGSDREQ
System Dumps Suppressed	The number of system dumps suppressed. Source field: SDGSDSUP
Transaction Dumps	The number of transaction dumps taken. Source field: SDGTDREQ
Transaction Dumps Suppressed	The number of transaction dumps suppressed. Source field: SDGTDSUP

TCP/IP reports

There are two TCP/IP reports, TCP/IP report, and TCP/IP services report.

TCP/IP report

The TCP/IP report is produced using a combination of **EXEC CICS INQUIRE TCPIP** and **EXEC CICS EXTRACT STATISTICS TCPIP** commands. The statistics data is mapped by the **DFHSOGDS DSECT**.

Table 280. Fields in the TCP/IP report

Field Heading	Description
TCP/IP Status	Indicates the current status of TCP/IP for this CICS system. Source field: **EXEC CICS INQUIRE TCPIP OPENSTATUS()**
SSLCACHE setting	Indicates the setting for the SSLCACHE system initialization parameter, which specifies whether SSL is to use the local or sysplex caching of session ids. Source field: SOG_SSLCACHE
Active SSL TCBs	The number of S8 TCBs in the SSL pool. Source field: INQUIRE DISPATCHER ACTSSLTCBS()
Maximum SSL TCBs (MAXSSLTCBS)	The maximum number of S8 TCBs allowed in the SSL pool, as specified by the MAXSSLTCBS system initialization parameter. Source field: INQUIRE DISPATCHER MAXSSLTCBS()
Max IP sockets (MAXSOCKETS) limit	The maximum number of IP sockets that can be managed by the CICS sockets domain. Source field: SOG-MAXSOCKETS-LIMIT
Number of times the MAXSOCKETS limit was reached	The number of times the maximum number of IP sockets limit (MAXSOCKETS) was reached. Source field: SOG-TIMES-AT-MAXSOCKETS

Table 280. Fields in the TCP/IP report (continued)

Field Heading	Description
Current Active IP sockets	The current number of IP sockets managed by the CICS sockets domain. Source field: **EXEC CICS INQUIRE TCPIP ACTSOCKETS()**
Current number of inbound sockets	The current number of inbound sockets. Source field: SOG-CURR-INBOUND-SOCKETS
Peak number of inbound sockets	The peak number of inbound sockets. Source field: SOG-PEAK-INBOUND-SOCKETS
Current number of non-persistent outbound sockets	The current number of non-persistent outbound sockets. Source field: SOG-CURR-OUTB-SOCKETS
Peak number of non-persistent outbound sockets	The peak number of non-persistent outbound sockets. Source field: SOG-PEAK-OUTB-SOCKETS
Current number of persistent outbound sockets	The current number of persistent outbound sockets. Source field: SOG-CURR-PERS-OUTB-SOCKETS
Peak number of persistent outbound sockets	The peak number of persistent outbound sockets. Source field: SOG-PEAK-PERS-OUTB-SOCKETS
Number of inbound sockets created	The total number of inbound sockets created. Source field: SOG-INBOUND-SOCKETS-CREATED
Number of outbound sockets created	The total number of outbound sockets created. Source field: SOG-OUTBOUND-SOCKETS-CREATED
Number of outbound sockets closed	The total number of outbound sockets closed. Source field: SOG-OUTBOUND-SOCKETS-CLOSED
Total number of inbound and outbound sockets created	The total number of inbound and outbound sockets created. Source field: SOG-INBOUND-SOCKETS-CREATED + SOG-OUTBOUND-SOCKETS-CREATED
Number of create socket requests delayed by MAXSOCKETS	The number of create socket requests that were delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-DELAYED-AT-MAX-SOCKETS
Total MAXSOCKETS delay time	The total time that create socket requests were delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-QTIME-AT-MAX-SOCKETS
Average MAXSOCKETS delay time	The average time that a create socket request was delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-QTIME-AT-MAX-SOCKETS / SOG-DELAYED-AT-MAX-SOCKETS
Number of create requests that timed-out at MAXSOCKETS	The number of create socket requests that were timed out while delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-TIMEDOUT-AT-MAXSOCKETS

Table 280. Fields in the TCP/IP report (continued)

Field Heading	Description
Current create socket requests delayed by MAXSOCKETS	The current number of create socket requests delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURR-DELAYED-AT-MAX
Peak create socket requests delayed by MAXSOCKETS	The peak number of create socket requests delayed because the system had reached the MAXSOCKETS limit. Source field: SOG-PEAK-DELAYED-AT-MAX
Total delay time for current create requests delayed	The total delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURRENT-QTIME-AT-MAX
Average delay time for current create requests delayed	The average delay time for the create socket requests that are currently delayed because the system is at the MAXSOCKETS limit. Source field: SOG-CURRENT-QTIME-AT-MAX / SOG-CURR-DELAYED-AT-MAX
Performance tuning for HTTP connections	Indicates whether performance tuning for HTTP connections will take place. Source field: SOG-SOTUNING
Socket listener pausing listening for HTTP connections	Indicates whether the listener has paused listening for HTTP connection requests because the number of tasks in the region has reached the limit for accepting new HTTP connection requests. Source field: SOG-PAUSING-HTTP-LISTENING
Number of times listener notified at task accept limit	The number of times the listener has been notified that the number of tasks in the region has reached the limit for accepting new HTTP connection requests. Source field: SOG-TIMES-AT-ACCEPT-LIMIT
Last time listener paused listening for HTTP connections	The last time the socket listener paused listening for HTTP connection requests because the number of tasks in the region had reached the limit for accepting new HTTP connection requests. The DFHSTUP report expresses this time as day/month/year hours:minutes:seconds:decimals; however, the DSECT field contains the time as a store clock (STCK) value in local time. If the DFHSTUP report shows the date and time as --/--/---- --:--:--:---- then that indicates that the listener has never paused listening for HTTP connection requests since the statistics were last reset Source field: SOG-TIME-LAST-PAUSED-LISTENING
Region stopping HTTP connection persistence	Indicates whether the region is stopping HTTP connection persistence because the number of tasks in the region has exceeded the limit. Source field: SOG-STOPPING-PERSISTENCE
Number of times region stopped HTTP connection persistence	The number of times the region took action to stop HTTP connection persistence because the number of tasks in the region has exceeded the limit. Source field: SOG-TIMES-STOPPED-PERSISTENT
Last time region stopped HTTP connection persistence	The last time the region took action to stop HTTP connection persistence because the number of tasks in the region has exceeded the limit. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that HTTP connection persistence has not been stopped since the statistics were last reset. Source field: SOG-TIME-LAST-STOPPED-PERSIST

Table 280. Fields in the TCP/IP report (continued)

Field Heading	Description
Number of persistent connections made non-persistent	The number of times a persistent HTTP connection was made non-persistent because the number of tasks in the region has exceeded the limit. Source field: SOG-TIMES-MADE-NON-PERSISTENT
Number of times disconnected a connection at max uses	The number of times a persistent HTTP connection was disconnected because the number of uses had exceeded the limit. Source field: SOG-TIMES-CONN-DISC-AT-MAX

TCP/IP services report

The TCP/IP services report is produced using a combination of **EXEC CICS INQUIRE TCPIPSERVICE** and **EXEC CICS EXTRACT STATISTICS TCPIPSERVICE** commands. The statistics data is mapped by the **DFHSORDS DSECT**.

Table 281. Fields in the TCP/IP Services report

Field Heading	Description
TCPIPSERVICE Name	The name of the TCP/IP service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE()**
TCPIPSERVICE Open Status	The current status of this TCP/IP service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() OPENSTATUS**(*cvda*)
Open Date and Time	The date and time when this TCP/IP service was opened. Source field: SOR-OPEN-LOCAL
TCPIPSERVICE Protocol	The protocol being used for this service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() PROTOCOL**(*cvda*)
TCPIPSERVICE Port	The number of the port on which CICS is listening on behalf of this service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() PORT()**
TCPIPSERVICE Host	The host name of the remote system or its IP address. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() HOST()**
TCPIPSERVICE IP Family	The address format of the address returned in the TCPIPSERVICE IP Resolved Address field. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() IPFAMILY**(*cvda*)
TCPIPSERVICE IP Resolved Address	The IPv4 or IPv6 resolved address of the host. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() IPRESOLVED()**
TCPIPSERVICE Transaction ID	The name of the transaction to be started to process a new request. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() TRANSID()**
TCPIPSERVICE Backlog	The port backlog setting for this TCP/IP service, which controls the number of requests that TCP/IP queues for this port before it starts to reject incoming requests. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() BACKLOG()**
TCPIPSERVICE URM	The name of the service user-replaceable module (URM) to be invoked by the attached task. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() TSQPREFIX**

Table 281. Fields in the TCP/IP Services report (continued)

Field Heading	Description
TCPIPSERVICE Maxdata	The setting for the maximum length of data that can be received by CICS as an HTTP server. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() MAXDATALEN()**
TCPIPSERVICE SSL Type	The level of secure sockets being used for the service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() SSLTYPE**(*cvda*)
TCPIPSERVICE Authenticate	The authentication requested for clients using this service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() AUTHENTICATE**(*cvda*)
TCPIPSERVICE Privacy	The level of SSL encryption required for inbound connections to this service. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() PRIVACY**(*cvda*)
TCPIPSERVICE Attachsec	For ECI over TCP/IP services, the level of attach-time security used by connections to CICS clients. Source field: **EXEC CICS INQUIRE TCPIPSERVICE() ATTACHSEC**(*cvda*)
Current Connections	The current number of connections for this TCP/IP service. Source field: SOR-CURRENT-CONS
Peak Connections	The peak number of connections for this TCP/IP service. Source field: SOR-PEAK-CONS
Transactions Attached	The total number of transactions attached for this TCP/IP service. Source field: SOR-TRANS-ATTACHED
Send requests	The number of send requests issued for the TCP/IP service. Source field: SOR-SENDS
Total Bytes Sent	The total number of bytes per send request for the TCP/IP service. Source field: SOR-BYTES-SENT
Receive requests	The number of receive requests issued for the TCP/IP service. Source field: SOR-RECEIVES
Total Bytes Received	The total number of bytes per receive request for the TCP/IP service. Source field: SOR-BYTES-RECEIVED
Maximum Persistent Connections	The maximum number of persistent connections from web clients that the CICS region accepts at any one time. Source field: SOR-TCPIPS-MAX-PERSIST
Non-Persistent Connections	The number of connections where CICS did not allow the web client to have a persistent connection. Source field: SOR-TCPIPS-NON-PERSIST

Temporary storage reports

There are five temporary storage reports, Temporary Storage report, Temporary Storage Main - Storage Subpools report, Temporary Storage Models report, Temporary Storage Queues report, and Temporary Storage Queues by shared TS Pool report.

Temporary Storage report

The Temporary Storage report is produced using the **EXEC CICS EXTRACT STATISTICS TSQUEUE** command. The statistics data is mapped by the **DFHTSGDS DSECT**.

Table 282. Fields in the Temporary Storage report

Field Heading	Description
Put/Putq main storage requests	The number of records that application programs wrote to main temporary storage. Source field: TSGSTA5F
Get/Getq main storage requests	The number of records that application programs obtained from main temporary storage. Source field: TSGNMG
Current TSMAINLIMIT setting	The current limit for the amount of storage that CICS makes available for data in main temporary storage. This amount is expressed in KB. Source field: (TSGTSMLM / 1024)
Times at TSMAINLIMIT	The number of times that main temporary storage use attempted to exceed the limit for the amount of storage allowed for data. Source field: TSGTSLHT
Current storage used for TSMAINLIMIT	The amount of storage that is currently in use for data in main temporary storage. This amount is expressed in KB. Source field: (TSGTSMUS / 1024)
Peak storage used for TSMAINLIMIT	The peak amount of storage that was used for data in main temporary storage. This amount is expressed in KB. Source field: (TSGTSMAX / 1024)
Number of queues auto deleted	The number of temporary storage queues that CICS has deleted automatically by using the clean up task. Source field: TSGTSQDL
Count of clean up task runs	The number of times that the clean up task, which deletes eligible temporary storage queues automatically, has run. Source field: TSGTSCTR
Put/Putq auxiliary storage requests	The number of records that application programs wrote to auxiliary temporary storage. Source field: TSGSTA7F
Get/Getq auxiliary storage requests	The number of records that application programs obtained from auxiliary temporary storage. Source field: TSGNAG
Times temporary storage queue created	The number of times that CICS created individual temporary storage queues. Source field: TSGSTA3F
Peak temporary storage queues in use	The peak number of temporary storage queue names in use at any one time. Source field: TSGQNUMH
Current temporary storage queues in use	The current number of temporary storage queue names in use. Source field: TSGQNUM

Table 282. Fields in the Temporary Storage report (continued)

Field Heading	Description
Items in longest queue	The peak number of items in any one temporary storage queue, up to a maximum of 32767. Source field: TSGQINH
Control interval size	The size of the VSAM 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 control intervals (CIs) permits more data to be transferred at one time, resulting in less system overhead. Source field: TSGCSZ
Control intervals in the DFHTEMP data set	The number of control intervals (CIs) available for auxiliary temporary 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: TSGNCI
Peak control intervals in use	The peak number of control intervals (CIs) that contain active data. Source field: TSGNCIAH
Current control intervals in use	The current number of control intervals (CIs) that contain active data. Source field: TSGNCIA
Available bytes per control interval	The number of bytes available for use in the temporary storage data set control interval. Source field: TSGNAVB
Segments per control interval	The number of segments available in each temporary storage data set control interval. Source field: TSGSPCI
Bytes per segment	The number of bytes per segment of the temporary storage data set. Source field: TSGBPSEG
Writes bigger than control interval size	The 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. Source field: TSGSTABF
Largest record length written	The size, expressed in bytes, of the longest record written to the temporary storage data set. Source field: TSGLAR
Times auxiliary storage exhausted	The number of situations where one or more transactions might have been suspended because of a NOSPACE condition, or might have been forced to end abnormally (by using a HANDLE CONDITION NOSPACE command). If statistics are present for this field, increase the size of the temporary storage data set. Source field: TSGSTA8F
Number Temporary Storage compressions	The number of times that the temporary storage buffers were compressed. Source field: TSGSTA9F

Table 282. Fields in the Temporary Storage report (continued)

Field Heading	Description
Put auxiliary / compression ratio	Ratio of temporary storage put auxiliary requests to temporary storage compressions. This ratio should be as high as possible to minimize compressions. Source field: (TSGSTA7F / TSGSTA9F)
Temporary storage strings	The number of temporary storage strings specified in the TS= system initialization parameter, or in the overrides. The number of strings allocated might exceed the number requested. Source field: TSGNVCA
Peak Temporary storage strings in use	The peak number of concurrent input/output operations. If this is significantly less than the number specified in the system initialization table (SIT), consider reducing the SIT value to approach this number. Source field: TSGNVCAH
Temporary storage string waits	The number of input/output requests that were queued because no strings were available. If the number of strings is the same as the number of buffers, this number is zero. If this number is a high percentage (over 30%) of the total number of input/output requests (for this purpose, the sum of TSGTWTN, Buffer writes, and TSGTRDN, Buffer reads), consider increasing the number of strings initially allocated. Source field: TSGVWTN
Peak users waiting on string	The peak number of input/output requests that were queued at any one time because all strings were in use. Source field: TSGVUWTH
Current users waiting on string	The current number of input/output requests that are queued because all strings are in use. Source field: TSGVUWT
Temporary storage buffers	The number of temporary storage buffers specified in the TS= system initialization parameter, or in the overrides. The number of buffers allocated might exceed the number requested. Source field: TSGNBCA
Temporary storage buffer waits	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TSGBWTN
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TSGBUWTH
Current users waiting on buffer	The current number of requests queued because no buffers are available. Source field: TSGBUWT
Temporary storage buffer reads	The number of times a control interval (CI) must be read from disk. To decrease this activity, increase the buffer allocation. Source field: TSGTRDN

Table 282. Fields in the Temporary Storage report (continued)

Field Heading	Description
Temporary storage buffer writes	The number of WRITEs to the temporary storage data set. This includes both WRITEs required for recovery (see Forced writes for recovery) and WRITEs required when the buffer is needed to accommodate another control interval (CI). To minimize input/output activity caused by the second situation, increase buffer allocation. Source field: TSGTWTN
Forced buffer writes for recovery	The subset of the total number of WRITEs caused by recovery being specified for queues. This input/output activity is not affected by buffer allocation. Source field: TSGTWTNR
Format writes	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 data set. A formatted write is attempted only if the current number of CIs available in the auxiliary data set have all been used. Source field: TSGTWTNF
I/O errors on the DFHTEMP data set	The number of input/output errors that occurred on the temporary storage data set. Normally, this number should be zero. If it is not, inspect the CICS and VSAM messages to determine the cause. Source field: TSGSTAAF
Shared Pools defined	The number of unique shared TS queue pools defined to CICS. Source field: TSGSHPDF
Shared Pools currently connected	The number of the shared TS pools that this CICS region is connected to. Source field: TSGSHPCN
Shared temporary storage read requests	The number of TS READQs from the Shared TS Queue pool of TS queues. Source field: TSGSHRDS
Shared temporary storage write requests	The number of TS WRITEQs to the Shared TS Queue pool of TS queues. Source field: TSGSHWTS
Storage Subpool Location	Storage location of the TSBUFFRS storage subpool. Source field: SMDDSANAME
Getmain Requests	The number of getmain requests issued for this TSBUFFRS storage subpool. Source field: SMDGMREQ
Freemain Requests	The number of freemain requests issued for this TSBUFFRS storage subpool. Source field: SMDFMREQ
Current Elements	The number of elements remaining after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests for this TSBUFFRS storage subpool. Source field: SMDCELEM
Current Element Storage	The amount of storage in bytes of the current elements. Source field: SMDCES

Table 282. Fields in the Temporary Storage report (continued)

Field Heading	Description
Current Page Storage	The current amount of page storage in kilobytes for this TSBUFFRS storage subpool. Source field: SMDCPS
% of ECDSA	The current element storage of the TSBUFFRS storage subpool as a percentage of the ECDSA in which it resides. Source field: ((SMDCPS / ecdsasize) * 100)
Peak Page Storage	The peak amount of page storage in kilobytes for this TSBUFFRS storage subpool. Source field: SMDHWMPS

Temporary Storage Main — Storage Subpools report

The Temporary Storage Main — Storage Subpools report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command. The statistics data is mapped by the **DFHSMDDS**.

The statistics data is mapped by the DFHSMDDS DSECT.

Table 283. Fields in the Temporary Storage Main — Storage Subpools report

Field Heading	Description
Subpool Name	The name of the temporary storage main subpool. Source field: SMDSPN
Location	The abbreviated name of the CICS dynamic storage area in which the subpool resides. ???? means that there has been no temporary storage main activity for this subpool. Source field: SMDDSANAME
Access	The storage key of the subpool. This can be either CICS (key 8) or USER (key 9). ???? means that there has been no temporary storage main activity for this subpool. Source field: SMDACCESS
Initial Free	The total number of kilobytes of the elements that are initially allocated when the subpool is preallocated. Source field: SMDIFREE
Getmain Requests	The number of GETMAIN requests issued for this subpool. Source field: SMDGMREQ
Freemain Requests	The number of FREEMAIN requests issued for this subpool. Source field: SMDFMREQ
Current Elements	The number of elements remaining after FREEMAIN requests; that is, it is the difference between the number of GETMAIN and FREEMAIN requests. Source field: SMDCELEM
Current Element Stg	The amount of storage in bytes of the current elements. Source field: SMDCES

Table 283. Fields in the Temporary Storage Main — Storage Subpools report (continued)

Field Heading	Description
Current Page Stg	The current amount of page storage in megabytes for this subpool. Source field: SMDCPS
% of DSA	The current element storage of the subpool as a percentage of the DSA in which it resides. Source field: ((SMDCPS / dsasize) * 100)
Peak Page Stg	The peak amount of page storage in kilobytes for this subpool. Source field: SMDHWMPS

Temporary Storage Models report

The Temporary Storage Models report is produced using the **EXEC CICS INQUIRE TSMODEL** command.

Table 284. Fields in the Temporary Storage Models report

Field Heading	Description
TSModel Name	The name of the temporary storage model. Source field: **EXEC CICS INQUIRE TSMODEL()**
Tsmodel Prefix	The prefix for this temporary storage model. Source field: **EXEC CICS INQUIRE TSMODEL() PREFIX**
Tsmodel Location	The location where queues matching this temporary storage model are to be stored. Source field: **EXEC CICS INQUIRE TSMODEL() LOCATION**(*cvda*)
Tsmodel Poolname	The name of the shared pool for this temporary storage model. Source field: **EXEC CICS INQUIRE TSMODEL() POOLNAME**
Recoverable	The recovery status for this temporary storage model. Source field: **EXEC CICS INQUIRE TSMODEL() RECOVSTATUS**(*cvda*)
Expiry Interval	The expiry interval for temporary storage queues that are associated with this temporary storage model. Source field: **EXEC CICS INQUIRE TSMODEL() EXPIRYINTMIN**

Temporary Storage Queues report

The Temporary Storage Queues report is produced using the **EXEC CICS INQUIRE TSQUEUE** command.

Table 285. Fields in the Temporary Storage Queues report

Field Heading	Description
Tsqueue Name	The name of the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME()**
Tsqueue Location	Indicates where the temporary storage queue resides. Source field: **EXEC CICS INQUIRE TSQNAME() LOCATION**(*cvda*)

Table 285. Fields in the Temporary Storage Queues report (continued)

Field Heading	Description
Number of Items	The number of items in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() NUMITEMS()**
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() MINITEMLEN()**
Max Item Length	The length of the largest item in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()**
Tsqueue Flength	The total length of all the items in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() FLENGTH()**
Tranid	The name of the transaction which created the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() TRANSID()**
Lastused Interval	The time interval since the temporary storage queue was last referenced. Source field: **EXEC CICS INQUIRE TSQNAME() LASTUSEDINT()**
Recoverable	Indicates whether the temporary storage queue is recoverable. Source field: **EXEC CICS INQUIRE TSQNAME() RECOVSTATUS()**
Expiry Interval	The expiry interval for this temporary storage queue, as defined in its TSMODEL resource definition at the time that the queue was created. Source field: **EXEC CICS INQUIRE TSMODEL() EXPIRYINTMIN()**

Temporary Storage Queues by Shared TS Pool report

The Temporary Storage Queues by Shared TS Pool report shows temporary storage queues that are in shared TS Pools on the TS Pool servers. These temporary storage queues might or might not currently be in the address space of your system. If they are not in the address space of your system, they are not shown on the other temporary storage queue reports.

The report is produced using a combination of the **EXEC CICS INQUIRE TSPOOL** and **EXEC CICS INQUIRE TSQUEUE** commands.

Table 286. Fields in the Tsqueue by Shared TS Pool report

Field Heading	Description
Shared TS Pool Name	The name of the shared temporary storage pool. Source field: **EXEC CICS INQUIRE TSPOOL()**
Connection Status	Indicates the connection status of the pool. Source field: **EXEC CICS INQUIRE TSPOOL() CONNSTATUS**(*cvda*)
TSQueue Name	The name of the temporary storage queue in this pool. Source field: **EXEC CICS INQUIRE TSQNAME()**
Number of Items	The number of items in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() NUMITEMS()**

Table 286. Fields in the Tsqueue by Shared TS Pool report (continued)

Field Heading	Description
Min Item Length	The length of the smallest item in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() MINITEMLEN()**
Max Item Length	The length of the largest item in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() MAXITEMLEN()**
Tsqueue Flength	The total length of all the items in the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() FLENGTH()**
Tranid	The name of the transaction which created the temporary storage queue. Source field: **EXEC CICS INQUIRE TSQNAME() TRANSID()**
Lastused Interval	The time interval since the temporary storage queue was last referenced. Source field: **EXEC CICS INQUIRE TSQNAME() LASTUSEDINT()**

Terminal Autoinstall and z/OS Communications Server report

The Terminal Autoinstall and z/OS Communications Server Report shows information and statistics about the status of terminal autoinstall - local terminals, and terminal autoinstall - shipped terminals. The report also shows the current status of the connection between CICS and the z/OS Communications Server, storage usage, generic resource usage and persistent session statistics.

The Terminal Autoinstall and z/OS Communications Server Reports are produced using a combination of the **EXEC CICS INQUIRE AUTOINSTALL**, **INQUIRE VTAM**, and the **EXEC CICS COLLECT STATISTICS AUTOINSTALL**, and Communications Server commands. The statistics data is mapped by the DFHA03DS, and DFHA04DS DSECTs.

Note: VTAM is a previous name for z/OS Communications Server.

Table 287. Fields in the Terminal Autoinstall report

Field Heading	Description
Terminal Autoinstall Status	Indicates the current status of terminal autoinstall. Source field: **EXEC CICS INQUIRE AUTOINSTALL ENABLESTATUS**(*cvda*)
Bridge Autoinstall	Indicates the current status of autoinstall for bridge facilities. Source field: **EXEC CICS INQUIRE AUTOINSTALL AIBRIDGE**(*cvda*)
Console Autoinstall	Indicates the current status of autoinstall for consoles. Source field: **EXEC CICS INQUIRE AUTOINSTALL CONSOLES**(*cvda*)
Autoinstall Program	The name of the user-replaceable terminal autoinstall model definition program. Source field: **EXEC CICS INQUIRE AUTOINSTALL PROGRAM()**
Current Autoinstall Requests	The number of autoinstall requests currently being processed. Source field: **EXEC CICS INQUIRE AUTOINSTALL CURREQS()**
Peak Autoinstall Requests	The maximum number of autoinstall requests that can be processed concurrently. Source field: **EXEC CICS INQUIRE AUTOINSTALL MAXREQS()**

Table 287. Fields in the Terminal Autoinstall report (continued)

Field Heading	Description
Autoinstalls Attempted	The number of terminal autoinstalls attempted. Source field: A04VADAT
Autoinstalls Rejected	The number of terminal autoinstalls rejected. Source field: A04VADRJ
Autoinstalls Deleted	The number of autoinstalled terminals deleted. Source field: A04VADLO
Peak Concurrent Autoinstalls	The peak number of autoinstall requests processed concurrently. Source field: A04VADPK
Times Peak Concurrent reached	The number of times the peak autoinstall requests was reached. Source field: A04VADPX
Times SETLOGON HOLD issued	The number of times the SETLOGON HOLD command was issued to prevent further logon requests. Source field: A04VADSH
Number of Queued Logons	The number of autoinstall attempts that were queued for logon because the delete was in progress for the same terminal. Source field: A04VADQT
Peak Number of Queued Logons	The peak number of autoinstall attempts that were queued for logon. Source field: A04VADQK
Times Peak Queued Logons reached	The number of times the peak number of autoinstall attempts that were queued for logon was reached. Source field: A04VADQX
Delete shipped definitions interval	The current delete redundant shipped terminal definitions interval. Source field: A04RDINT
Delete shipped definitions Idle time	The current minimum time that an inactive shipped terminal definition must remain installed in this region before it becomes eligible for deletion. Source field: A04RDIDL
Shipped remote terminals built	The total number of shipped terminal definitions that have been installed in this region. Source field: A04SKBLT
Shipped remote terminals installed	The number of shipped terminal definitions currently installed in this region. Source field: A04SKINS
Shipped remote terminals deleted	The number of shipped terminal definitions deleted from this region. Source field: A04SKDEL
Times remote delete interval expired	The number of times the remote delete interval has expired. Source field: A04TIEXP
Remote terminal deletes received	The number of remote delete requests received by this region. Source field: A04RDREC

Table 287. Fields in the Terminal Autoinstall report (continued)

Field Heading	Description
Remote terminal deletes issued	The number of remote delete requests issued by this region. Source field: A04RDISS
Successful remote terminal deletes	The number of shipped terminal definitions deleted in this region by remote delete requests. Source field: A04RDDEL
Current idle terminals awaiting reuse	The current number of remote terminal definitions that are idle and are awaiting reuse. Source field: A04CIDCT
Current idle time awaiting reuse	The total time that the current number of remote terminal definitions that are awaiting reuse have been idle. Source field: A04CIDLE
Current maximum idle time awaiting reuse	The current maximum time that a remote terminal definition that is awaiting reuse has been idle. Source field: A04CMAXI
Total idle terminal count awaiting reuse	The total number of remote terminal definitions that have been idle and awaited reuse. Source field: A04TIDCT
Total idle time awaiting reuse	The total time that the total number of remote terminal definitions that awaited reuse were idle. Source field: A04TIDLE
Average idle time awaiting reuse	The average time that the remote terminal definitions were idle awaiting reuse. Source field: A04TIDLE / A04TIDCT
Maximum idle time awaiting reuse	The maximum time a shipped terminal definition has been idle awaiting reuse. Source field: A04TMAXI

Table 288. Fields in the z/OS Communications Server report

Field Heading	Description
VTAM open status	The current status of the connection between CICS and the Communications Server. Source field: **EXEC CICS INQUIRE VTAM OPENSTATUS**(*cvda*)
Dynamic open count	The number of times the Communications Server ACB was dynamically opened. Source field: A03DOC
VTAM Short-on-Storage	The number of times that the Communications Server indicated that there was a temporary Communications Server storage problem. Source field: A03VTSOS
MAX RPLs	The maximum number of receive-any request parameter lists (RPLs) that were posted by the Communications Server on any one dispatch of CICS terminal control. Source field: A03RPLX

Table 288. Fields in the z/OS Communications Server report (continued)

Field Heading	Description
Times at MAX RPLs	The number of times the maximum number of receive-any request parameter lists (RPLs) was reached. Source field: A03RPLXT
Current LUs in session	The current number of LUs in session. Source field: A03LUNUM
Peak LUs in session	The peak number of LUs in session. Source field: A03LUHWM
Generic Resource name	The name of the generic resource group which this CICS region requested registration to the Communications Server. Source field: **EXEC CICS INQUIRE VTAM GRNAME()**
Generic Resource status	Indicates the status of generic resource registration. Source field: **EXEC CICS INQUIRE VTAM GRSTATUS**(*cvda*)
Persistent Session Type	The setting for Communications Server persistent sessions support in the CICS region, as specified by the system initialization parameter PSTYPE. The settings are as follows: • SNPS - single-node persistent sessions • MNPS - multinode persistent sessions • NOPS - persistent sessions support is not used Source field: A03PSTYP
Persistent Session Interval	The time for which persistent sessions are retained if a failure occurs, as specified by the system initialization parameter PSDINT. Source field: A03PSDIN
Persistent Session Inquire count	The number of times CICS issued VTAM INQUIRE OPTCD=PERSESS to inquire on the number of persistent sessions. Source field: A03PSIC
Persistent Session NIB count	The number of Communications Server sessions that persisted. Source field: A03PSNC
Persistent Session Opndst count	The number of persisting sessions that were successfully restored. Source field: A03PSOC
Persistent Session Unbind count	The number of persisting sessions that were stopped. Source field: A03PSUC
Persistent Session Error count	The number of persisting sessions that were already unbound when CICS tried to restore them. Source field: A03PSEC

Tsqueue Totals report

The Tsqueue Totals report shows totals that are calculated from data gathered using the **EXEC CICS INQUIRE TSQUEUE** command.

Table 289. Fields in the Tsqueue Totals report

Field Heading	Description
Current temporary storage queues	The total number of temporary storage queues currently in use.
Current auxiliary temporary storage queues	The total number of temporary storage queues currently in auxiliary storage. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Current items in auxiliary temporary storage queues	The total number of items in temporary storage queues currently in auxiliary storage. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Average items per auxiliary temporary storage queue	The average number of items in each temporary storage queue currently in auxiliary storage. Source field: Current items in auxiliary temporary storage queues / Current auxiliary temporary storage queues
Current main temporary storage queues	The total number of temporary storage queues currently in main storage. Source field: EXEC CICS INQUIRE TSQNAME() LOCATION(cvda)
Current items in main temporary storage queues	The total number of items in temporary storage queues currently in main storage. Source field: EXEC CICS INQUIRE TSQNAME() NUMITEMS()
Average items per main temporary storage queue	The average number of items in each temporary storage queue currently in main storage. Source field: Current items in main temporary storage queues / Current main temporary storage queues

Trace Settings report

The Trace Settings report is produced using the **EXEC CICS INQUIRE TRACEDEST**, **EXEC CICS INQUIRE TRACEFLAG**, **EXEC CICS INQUIRE TRACETYPE**, **EXEC CICS INQUIRE JVMPOOL**, **EXEC CICS INQUIRE TRANSACTION**, and **EXEC CICS EXTRACT STATISTICS TRANSACTION** commands.

Table 290. Fields in the Trace Settings report

Field Heading	Description
Trace Settings	
Internal Trace Status	The status of CICS internal trace (started or stopped). Source field: EXEC CICS INQUIRE TRACEDEST INTSTATUS
Internal Trace Table Size	The size of the table that holds internal trace entries. The table wraps when it is full. Source field: EXEC CICS INQUIRE TRACEDEST TABLESIZE
Auxiliary Trace Status	The status of CICS auxiliary trace (started or stopped). Source field: EXEC CICS INQUIRE TRACEDEST AUXSTATUS

Table 290. Fields in the Trace Settings report (continued)

Field Heading	Description
Auxiliary Trace Dataset	The current auxiliary trace data set. Source field: EXEC CICS INQUIRE TRACEDEST CURAUXDS
Auxiliary Switch Status	The status of the auxiliary switch, which determines what happens when the initial data set for auxiliary trace is full. Source field: EXEC CICS INQUIRE TRACEDEST SWITCHSTATUS
GTF Trace Status	The status of CICS GTF trace (started or stopped), that is, whether CICS is directing trace entries to the MVS Generalized Trace Facility (GTF). Source field: EXEC CICS INQUIRE TRACEDEST GTFSTATUS
Master System Trace Flag	The status of the system master trace flag, which governs whether CICS makes or suppresses standard trace entries. Source field: EXEC CICS INQUIRE TRACEFLAG SYSTEMSTATUS
Master User Trace Flag	The status of the user master trace flag, which governs whether non-exception user trace entries are recorded or suppressed. Source field: EXEC CICS INQUIRE TRACEFLAG SYSTEMSTATUS
VTAM Exit override	Indicates which invocations of the CICS z/OS Communications Server exits are being traced. Source field: EXEC CICS INQUIRE TRACEFLAG TCEXITSTATUS
JVM Trace Options	
Standard	The setting for standard tracing for this trace flag. Source field: EXEC CICS INQUIRE TRACETYPE COMPID(SJ) STANDARD
Special	The setting for special tracing for this trace flag. Source field: EXEC CICS INQUIRE TRACETYPE COMPID(SJ) SPECIAL
Option String	The JVM trace options for this trace flag. Source field: EXEC CICS INQUIRE JVMPOOL JVMLEVEL0TRACE, JVMLEVEL1TRACE, JVMLEVEL2TRACE, or JVMUSERTRACE
Component Trace Options	
Component	The name of the component for tracing. Source field: EXEC CICS INQUIRE TRACETYPE COMPID
Description	The description of the component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID
Standard	The active level of tracing for standard tracing for this component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID() STANDARD
Special	The active level of tracing for special tracing for this component. Source field: EXEC CICS INQUIRE TRACETYPE COMPID() SPECIAL
Transactions - Non-Standard Tracing	
Tran id	The name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION

Table 290. Fields in the Trace Settings report (continued)

Field Heading	Description
Tran Class	The transaction class in which the transaction is defined. Source field: XMRTCL
Program Name	The name of the program when the transaction was defined, or spaces if a program name was not supplied. Source field: XMMRPN
Tracing	The type of tracing to be done for tasks executing this transaction. Source field: EXEC CICS INQUIRE TRANSACTION() TRACING
Attach Count	The number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction runs. Source field: XMRAC
Restart Count	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	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	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	The number of times that this transaction definition has been used to attempt to start the transaction on a remote system. See additional information in "Transactions report." Source field: XMRRSC

Transaction reports

There are four transaction reports, Transactions report, Transaction Classes report, Transaction Manager report, and Transaction Totals report.

Transactions report

The Transactions report is produced using a combination of the **EXEC CICS INQUIRE TRANSACTION** and **EXEC CICS EXTRACT STATISTICS TRANSACTION** commands. The statistics data is mapped by the **DFHXMRDS**.

Table 291. Fields in the Transactions Report

Field Heading	Description
Tran id	The name of the transaction. Source field: EXEC CICS INQUIRE TRANSACTION
Tran Class	The name of the transaction class in which the transaction is defined. Source field: XMRTCL

Table 291. Fields in the Transactions Report (continued)

Field Heading	Description
Program Name	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	Where certain CICS control blocks will be located for the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATALOC
Task Data Key	The storage key in which CICS will obtain all storage for use by the transaction. Source field: EXEC CICS INQUIRE TRANSACTION TASKDATAKEY
Attach Count	The number of times that this transaction has been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction runs. Source field: XMRAC
Restart Count	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	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	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	The number of times that this transaction definition has been used to attempt to start the transaction on a remote system. (This might not necessarily be the same as the number of successful starts.) A Remote Start is only counted in the CICS region that initiates the process, and not in the remote system where the transaction runs. In some circumstances, the use of a transaction definition for a remote start is not counted. This includes the case where a transaction definition that specifies the local sysid or nothing as the REMOTESYSTEM value, is used to start a transaction in a remote system, with the remote system specified on the SYSID option of the START command. Source field: XMRRSC
Storage Viols	The number of times a storage violation has been detected for this transaction definition. Source field: XMRSVC

Transaction Classes report

The Transaction Classes report is produced using a combination of the **EXEC CICS INQUIRE TRANCLASS** and **EXEC CICS EXTRACT STATISTICS TRANCLASS** commands.

The statistics data is mapped by the DFHXMCDS DSECT.

Table 292. Fields in the Transaction Classes report pass 1

Field Heading	Description
Tclass Name	The name of the transaction class. Source field: EXEC CICS INQUIRE TRANCLASS()
Trans in Tclass	The number of transaction definitions that are defined to this transaction class. Source field: XMCITD
Attach in Tclass	The number of transaction attach requests for transactions in this transaction class. Source field: XMCTAT
Class Limit	The maximum number of transactions that may be concurrently active in this transaction class. Source field: XMCMXT
At Class Limit	The number of times that this transaction class has reached its transaction class limit. Source field: XMCTAMA
Current Active	The current number of transactions active in this transaction class. Source field: XMCCAT
Peak Active	The peak number of transactions active in this transaction class. Source field: XMCPAT
Current Queued	The current number of transactions that are currently queueing in this transaction class. Source field: XMCCQT
Peak Queued	The peak number of transactions that queued waiting to get into this transaction class. Source field: XMCPQT
Accept Immediate	The number of transactions that were accepted immediately into this transaction class. Source field: XMCAI
Accept Queued	The number of transactions that were queued before being accepted into this transaction class. Source field: XMCAAQ

Table 293. Fields in the Transaction Classes report pass 2

Field Heading	Description
Tclass Name	The name of the transaction class. Source field: EXEC CICS INQUIRE TRANCLASS()

Table 293. Fields in the Transaction Classes report pass 2 (continued)

Field Heading	Description
Trans in Tclass	The number of transaction definitions that are defined to this transaction class. Source field: XMCITD
Class Limit	The maximum number of transactions that may be concurrently active in this transaction class. Source field: XMCMXT
Purge Threshold	The queue limit purge threshold for this transaction class. Source field: XMCTH
At Purge Threshold	The number of times this transaction class has reached its queue limit purge threshold. Source field: XMCTAPT
Purged Immediate	The number of transactions that were purged immediately because the queue had already reached the purge threshold for this transaction class. Source field: XMCPI
Purged Queued	The number of transactions that have been purged while queueing to get into this transaction class. Source field: XMCPWQ
Total Queued	The total number of transactions that have become active but first queued to get into this transaction class. Source field: XMCTQ
Avg. Queue Time	The average queueing time for transactions that have become active but first queued to get into this transaction class. Source field: XMCTQTME / XMCTQ
Avg. Cur Queue Time	The average queueing time for those transactions that are currently queued waiting to get into this transaction class. Source field: XMCCQTME / XMCCQT

Transaction Manager report

The Transaction Manager report is produced using the **EXEC CICS EXTRACT STATISTICS TRANSACTION** command.

The statistics data is mapped by the DFHXMGDS DSECT.

Table 294. Fields in the Transaction Manager report

Field Heading	Description
Total Accumulated transactions so far	The total number of tasks that have accumulated so far. Source field: (XMGTNUM + XMGNUM)
Accumulated transactions (since reset)	The number of tasks that have accumulated since the last reset. Source field: XMGNUM
Transaction Rate per second	The number of transactions per second. Source field: (XMGNUM / Elapsed seconds since reset)

Table 294. Fields in the Transaction Manager report (continued)

Field Heading	Description
Maximum transactions allowed (MXT)	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
Time MXT last changed	The date and time when the maximum number of user transactions (MXT) was last set or changed dynamically. Source field: XMGLSMXT
Times at MXT	The number of times that the number of active user transactions equalled the specified maximum number of user transactions (MXT). Source field: XMGTAMXT
Time MXT last reached	The date and time when the number of active user transactions last equalled the specified maximum number of user transactions (MXT). Source field: XMGLAMXT
Current Active User transactions	The current number of active user transactions. Source field: XMGCAT
Currently at MXT	Whether the CICS region is currently at the specified maximum number of user transactions (MXT). Source field: XMGATMXT
Peak Active User transactions	The peak number of active user transactions reached. Source field: XMGPAT
Total Active User transactions	The total number of user transactions that have become active. Source field: XMGTAT
Time last transaction attached	The date and time when the last user transaction was attached. If the DFH0STAT report shows the date and time as --/--/---- --:--:--:---- then that indicates that a user transaction has not been attached since the statistics were last reset. Source field: XMGLTAT
Current Running transactions	The current number of Running transactions. Source field: EXEC CICS INQUIRE TASKLIST RUNNING
Current Dispatchable transactions	The current number of Dispatchable transactions. Source field: EXEC CICS INQUIRE TASKLIST DISPATCHABLE
Current Suspended transactions	The current number of Suspended transactions. Source field: EXEC CICS INQUIRE TASKLIST SUSPENDED
Current System transactions	The current number of system transactions. Source field: ((Running + Dispatchable + Suspended) - XMGCAT)
Transactions Delayed by MXT	The number of user transactions that had to queue for MXT reasons before becoming active, excluding those still waiting. Source field: XMGTDT

Table 294. Fields in the Transaction Manager report (continued)

Field Heading	Description
Total MXT Queueing Time	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	The average time spent waiting by those user transactions that had to wait for MXT reasons. Source field: (XMGTQTME / XMGTDT)
Current Queued User transactions	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
Peak Queued User transactions	The peak number of user transactions queuing for MXT reasons. **Note:** That this does not include transactions queued for Transaction Class. Source field: XMGPQT
Total Queueing Time for current queued	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	The average time spent waiting by those user transactions currently queued for MXT reasons. Source field: (XMGCQTME / XMGCQT)

Transaction Totals report

The Transactions Totals report is produced using the **EXEC CICS EXTRACT STATISTICS STORAGE** command.

The statistics data was mapped by the DFHSMSDS DSECT.

Table 295. Fields in the Transaction Totals report

Field Heading	Description
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	Indicates the combination of task data location and task data key for these transactions.
Subspace Usage	Indicates the type of subspace usage for these transaction definitions.
Transaction Count	The number of transaction definitions for this combination of isolate, task data location, task data key, and subspace usage.
Attach Count	The number of times that these transactions have been attached. If a transaction definition is used to start a transaction remotely, the transaction is included in the Attach Count for the region where the transaction runs.
Current Unique Subspace Users (Isolate=Yes)	The current number of tasks allocated a unique subspace. Source field: SMSUSSCUR

Table 295. Fields in the Transaction Totals report (continued)

Field Heading	Description
Peak Unique Subspace Users (Isolate=Yes)	The peak number of tasks allocated a unique subspace. Source field: SMSUSSHWM
Total Unique Subspace Users (Isolate=Yes)	The total number of tasks that have been allocated a unique subspace. Source field: SMSUSSCUM
Current Common Subspace Users (Isolate=No)	The current number of tasks allocated to the common subspace. Source field: SMSCSSCUR
Peak Common Subspace Users (Isolate=No)	The peak number of tasks allocated to the common subspace. Source field: SMSCSSHWM
Total Common Subspace Users (Isolate=No)	The total number of tasks that have been allocated to the common subspace. Source field: SMSCSSCUM

Transient data reports

There are three transient data reports, Transient Data report, Transient Data Queues report, and Transient Data Queue totals report.

Transient Data report

The Transient Data report is produced using the **EXEC CICS EXTRACT STATISTICS TDQUEUE** command. The statistics data is mapped by the **DFHTQGDS**.

Table 296. Fields in the Transient Data report

Field Heading	Description
Transient data reads	The number of times a CI has to be read from disk. Increasing the buffer allocation decreases this activity. Source field: TQGACTGT
Transient data writes	The number of WRITEs to the intrapartition transient data set. This includes both WRITEs needed for recovery 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: TQGACTPT
Transient data formatting writes	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: TQGACTFT
Control interval size	The size of the control interval, expressed in bytes. Source field: TQGACISZ
Control intervals in the DFHINTRA data set	The current number of control intervals active within the intrapartition data set, DFHINTRA. Source field: TQGANCIS
Peak control intervals used	The peak value of the number of control intervals concurrently active in the system. Source field: TQGAMXCI

Table 296. Fields in the Transient Data report (continued)

Field Heading	Description
Times NOSPACE on DFHINTRA occurred	The number of times that a NOSPACE condition has occurred. Source field: TQGANOSP
Transient data strings	The number of strings currently active. Source field: TQGSTSTA
Times Transient data string in use	The number of times a string was accessed. Source field: TQGSTNAL
Peak Transient data strings in use	The peak number of strings concurrently accessed in the system. Source field: TQGSMXAL
Times string wait occurred	The number of times that tasks had to wait because no strings were available. Source field: TQGSTNWT
Peak users waiting on string	The peak number of concurrent string waits in the system. Source field: TQGSMXWT
Transient data buffers	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: TQGANBFA
Times Transient data buffer in use	The number of times intrapartition buffers have been accessed. Source field: TQGATNAL
Peak Transient data buffers in use	The peak value of the number of concurrent intrapartition buffer accesses. Source field: TQGAMXAL
Peak buffers containing valid data	The peak number of intrapartition buffers that contain valid data. Source field: TQGAMXIU
Times buffer wait occurred	The number of times a request was queued because all buffers were allocated to other tasks. A buffer wait also occurs if the required control interval is already in a locked buffer, and therefore unavailable, even if there are other buffers available. Source field: TQGATNWT
Peak users waiting on buffer	The peak number of requests queued because no buffers were available. Source field: TQGAMXWT
I/O errors on the DFHINTRA data set	The number of input/output errors that have occurred on the DFHINTRA data set. Source field: TQGACTIO

Transient Data Queues report

The Transient Data Queues report is produced using a combination of the **EXEC CICS INQUIRE TDQUEUE** and **EXEC CICS EXTRACT STATISTICS TDQUEUE** commands. The statistics data is mapped by the **DFHTQRDS DSECT**.

Table 297. The Fields in the Transient Data Queue report

Field Heading	Description
Dest Id	The destination identifier (transient data queue name). Source field: **EXEC CICS INQUIRE TDQUEUE()**
Queue Type	The queue type, extrapartition, intrapartition, indirect or remote. Source field: **EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)**
Tdqueue Writes	The number of requests to write to the transient data queue. Source field: TQRWRITE
Tdqueue Reads	The number of requests to read from the transient data queue. Source field: TQRREAD
Tdqueue Deletes	The number of requests to delete from the transient data queue. Source field: TQRDELET
Indirect Name	The name of the indirect queue. Source field: TQRIQID
Remote System	The remote connection name (sysid) of the system for this queue. Source field: TQRRSYS
Remote Name	The remote queue name for this queue. Source field: TQRRQID
Current Items	The current number of items in this intrapartition queue. Source field: TQRCNITM
Peak Items	The peak number of items in this intrapartition queue. Source field: TQRPNITM
No.of triggers	The number of times a trigger transaction has been attached. Source field: TQRTRIGN
Trigger Level	The number of items that must be in this queue before automatic transaction initiation (ATI) occurs. Source field: TQRTRIGL
ATI Fcty	Indicates whether this queue has a terminal or session associated with it. Source field: **EXEC CICS INQUIRE TDQUEUE() ATIFACILITY(cvda)**
ATI Term	The name of the terminal or session associated with this queue. Source field: **EXEC CICS INQUIRE TDQUEUE() ATITERMID()**
ATI Tran	The name of the transaction to be attached when the trigger level for this queue is reached. Source field: TQRATRAN
ATI Userid	The user identifier associated with this queue. Source field: **EXEC CICS INQUIRE TDQUEUE() ATIUSERID()**

Transient Data Queue Totals report

The Transient Data Queues Totals report is produced using a combination of the **EXEC CICS INQUIRE TDQUEUE** and **EXEC CICS EXTRACT STATISTICS TDQUEUE** commands. The statistics data is mapped by the **DFHTQRDS DSECT**.

Table 298. Fields in the Transient Data Queue Totals report

Field Heading	Description
Tdqueue Type	The queue type, extrapartition, intrapartition, indirect, or remote. Source field: **EXEC CICS INQUIRE TDQUEUE() TYPE(cvda)**
No. of Tdqueues	The number of queues defined as this type.
Tdqueue Writes	The total number of requests to write to this type of transient data queue. Source field: TQRWRITE
Tdqueue Reads	The total number of requests to read from this type of transient data queue. Source field: TQRREADS
Tdqueue Deletes	The total number of requests to delete from this type of transient data queue. Source field: TQRDELET

URIMAP reports

There are two URIMAP reports, URIMAPs Global report, and URIMAPs report.

URIMAPs Global report

The URIMAPs Global report is produced using the **EXEC CICS EXTRACT STATISTICS URIMAP** command. The statistics data is mapped by the **DFHWBGDS DSECT**.

Table 299. Fields in the URIMAPs Global report

Field Heading	Description
URIMAP reference count	Number of times a search for a matching URIMAP definition was made. Source field: WBG-URIMAP-REFERENCE-COUNT
Entry point reference count	Number of times a search for a matching URIMAP definition that is defined as an application entry point was made. Source field: WBG-URIMAP-ENTRYPOINT-REF
Host/Path no match count	Number of times a search for a matching URIMAP definition was made, but no URIMAP definition with a matching host and path was found. Source field: WBG-URIMAP-NO-MATCH-COUNT
Host/Path match count	Number of times a search for a matching URIMAP definition was made, and a URIMAP definition with a matching host and path was found. Source field: WBG-URIMAP-MATCH-COUNT
Disabled	Number of times a URIMAP definition with a matching host and path was found, but the URIMAP definition was disabled. Source field: WBG-URIMAP-MATCH-DISABLED
Redirected	Number of times a URIMAP definition with a matching host and path was found, and the request was redirected. Source field: WBG-URIMAP-MATCH-REDIRECT

Table 299. Fields in the URIMAPs Global report (continued)

Field Heading	Description
Analyzer used	Number of times a URIMAP definition with a matching host and path was found, and the analyzer program associated with the TCPIPSERVICE definition was called. Source field: WBG-URIMAP-MATCH-ANALYZER
Static content delivered	Number of times a URIMAP definition with a matching host and path was found, and static content (document template or z/OS UNIX file) was delivered as a response. Source field: WBG-URIMAP-STATIC-CONTENT
Dynamic content delivered	Number of times a URIMAP definition with a matching host and path was found, and dynamic content (produced by an application program) was delivered as a response. Source field: WBG-URIMAP-DYNAMIC-CONTENT
PIPELINE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by a web service. Source field: WBG-URIMAP-PIPELINE-REQS
ATOMSERVICE requests	Number of times a URIMAP definition with a matching host and path was found, and the request was handled by an Atom service. Source field: WBG-URIMAP-ATOMSERV-REQS
Scheme (HTTP) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTP. Source field: WBG-URIMAP-SCHEME-HTTP
Scheme (HTTPS) requests	Number of times a URIMAP definition with a matching host and path was found, and the scheme was HTTPS (HTTP with SSL). Source field: WBG-URIMAP-SCHEME-HTTPS
Virtual host disabled count	Number of times a URIMAP definition with a matching host and path was found, but the virtual host was disabled. Source field: WBG-HOST-DISABLED-COUNT
Direct attach count	Number of requests that are processed by directly attached user task. Source field: WBG-URIMAP-DIRECT-ATTACH

URIMAPs report

The URIMAPs report is produced using a combination of **EXEC CICS INQUIRE URIMAP** and **EXEC CICS EXTRACT STATISTICS URIMAP RESID()** commands. The statistics data is mapped by the DFHWBRDS DSECT.

Table 300. Fields in the URIMAPs Report

Field Heading	Description
URIMAP Name	The name of the URIMAP definition. Source field: **EXEC CICS INQUIRE URIMAP**
URIMAP Enable Status	Whether the URIMAP definition is enabled, disabled, or unavailable because the virtual host of which it is a part has been disabled. Source field: **EXEC CICS INQUIRE URIMAP() ENABLESTATUS**

Table 300. Fields in the URIMAPs Report (continued)

Field Heading	Description
URIMAP Usage	The intended use of this URIMAP resource: **SERVER** The URIMAP resource is used to locate the resources for CICS to produce an HTTP response to the request identified by HOST and PATH. **CLIENT** The URIMAP resource is used to specify information for making an HTTP request from CICS as an HTTP client. **PIPELINE** The URIMAP resource is used to locate the resources for CICS to produce an XML response to the request identified by HOST and PATH. **ATOM** The URIMAP resource is used for an incoming request for data that CICS makes available as an Atom feed. **JVMSERVER** The URIMAP resource is used to map an inbound request from a web client to a servlet or JSP that is running in a JVM server. Source field: **EXEC CICS INQUIRE URIMAP() USAGE**
URIMAP Scheme	The scheme for the HTTP request, HTTP with SSL (HTTPS) or without SSL (HTTP). Source field: **EXEC CICS INQUIRE URIMAP() SCHEME**
URIMAP Authenticate	For USAGE(CLIENT), whether credentials (authentication information) are sent for outbound Web requests. Source field: **EXEC CICS INQUIRE URIMAP() AUTHENTICATE**
URIMAP Port	For USAGE(CLIENT), the port number used for the client connection. For USAGE(SERVER), the port number that is being used for the communication, even if PORT(NO) is specified on the URIMAP at define time. Source field: **EXEC CICS INQUIRE URIMAP() PORT()**
URIMAP Host	For USAGE(CLIENT), the host name of the target URL to which the HTTP request is to be sent. For any other usage type, the host name on the incoming HTTP request that is used to select this URIMAP definition. Source field: **EXEC CICS INQUIRE URIMAP() HOST()**
URIMAP IP Family	The address format of the address returned in URIMAP IP Resolved Address. Source field: **EXEC CICS INQUIRE URIMAP() IPFAMILY()**
URIMAP IP Resolved Address	The IPv4 or IPv6 resolved address of the host. Source field: **EXEC CICS INQUIRE URIMAP() IPRESOLVED()**
URIMAP Path	For USAGE(CLIENT), the path of the target URL to which the HTTP request is to be sent. For any other usage type, the path on the incoming HTTP request that is used to select this URIMAP definition. The PATH might end in an asterisk, meaning that it is generic, and matches any path with characters that are the same up to but excluding the asterisk. Source field: **EXEC CICS INQUIRE URIMAP() PATH**
TCPIPSERVICE name	The TCPIPSERVICE resource to which this URIMAP definition applies. Only requests received using this TCPIPSERVICE resource are matched to this URIMAP definition. If no TCPIPSERVICE resource is specified, the URIMAP definition applies to all incoming HTTP requests. Source field: **EXEC CICS INQUIRE URIMAP() TCPIPSERVICE**

Table 300. Fields in the URIMAPs Report (continued)

Field Heading	Description
WEBSERVICE name	The name of the WEBSERVICE resource definition for the web service that handles the incoming HTTP request. Source field: **EXEC CICS INQUIRE URIMAP() WEBSERVICE**
PIPELINE name	The name of the PIPELINE resource definition for the web service that handles the incoming HTTP request. Source field: **EXEC CICS INQUIRE URIMAP() PIPELINE**
ATOMSERVICE name	The name of the ATOMSERVICE resource definition for the Atom document. Source field: **EXEC CICS INQUIRE URIMAP() ATOMSERVICE**
Templatename	The name of a CICS document template, the contents of which are returned as the HTTP response. Source field: **EXEC CICS INQUIRE URIMAP() TEMPLATENAME**
zFS File	The name of a file in the z/OS UNIX System Services file system, the contents of which are returned as the HTTP response. Source field: **EXEC CICS INQUIRE URIMAP() HFSFILE**
Analyzer	Whether or not the analyzer associated with the TCPIPSERVICE definition is called to process the request. Source field: **EXEC CICS INQUIRE URIMAP() ANALYZERSTAT**
Converter	The name of a converter program that is used to transform the HTTP request into a form suitable for the application program specified in PROGRAM. Source field: **EXEC CICS INQUIRE URIMAP() CONVERTER**
Transaction ID	The name of the alias transaction that processes the incoming HTTP request. Source field: **EXEC CICS INQUIRE URIMAP() TRANSACTION**
Program name	The name of the application program that processes the incoming HTTP request. Source field: **EXEC CICS INQUIRE URIMAP() PROGRAM**
Redirection type	Whether or not matching requests are redirected, on a temporary or permanent basis. Source field: **EXEC CICS INQUIRE URIMAP() REDIRECTTYPE**
Location for redirection	An alternative URL to which the Web client is redirected, if redirection is specified. Source field: **EXEC CICS INQUIRE URIMAP() LOCATION**
URIMAP reference count	Number of times this URIMAP definition was referenced. Source field: WBR-URIMAP-REFERENCE-COUNT
Disabled	Number of times this host and path were matched, but the URIMAP definition was disabled. Source field: WBR-URIMAP-MATCH-DISABLED
Redirected	Number of times that this host and path were matched and the number of times that the request was redirected. Source field: WBR-URIMAP-MATCH-REDIRECT

Table 300. Fields in the URIMAPs Report (continued)

Field Heading	Description
Time out for pooled sockets	The time after which CICS discards pooled client HTTP connections created using this URIMAP resource if they are not reused. Source field: WBR-URIMAP-SOCKETCLOSE
Number of pooled sockets	Current number of open client HTTP connections held in the pool for reuse. Source field: WBR-URIMAP-SOCKPOOLSIZE
Peak number of pooled sockets	Peak number of open client HTTP connections held in the pool for reuse. Source field: WBR-URIMAP-SOCKPOOLSIZE-PEAK
Number of reclaimed sockets	Number of pooled connections that were closed in the pool by CICS because the CICS region had reached the MAXSOCKETS limit. Source field: WBR-URIMAP-SOCKETS-RECLAIMED
Number of timed out sockets	Number of pooled connections that were closed in the pool by CICS because they reached their timeout value without being reused. Source field: WBR-URIMAP-SOCKETS-TIMEDOUT

User Exit Programs report

The User Exit Programs report is produced from two tables. This report is produced using the **EXEC CICS INQUIRE EXITPROGRAM** command.

Table 301. Fields in the User Exit Programs report

Field Heading	Description
Program Name	The program name of the program that is enabled as an exit program by using the EXEC CICS ENABLE command. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The entry point name for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
Global Area Entry Name	The name of the exit program that owns the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAENTRYNAME()
Global Area Length	The length of the global work area associated with this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GALENGTH()
Global Area Use Count	The number of exit programs that are associated with the global work area owned by this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() GAUSECOUNT()
Number of Exits	The number of global user exit points at which this exit program is enabled. Source field: EXEC CICS INQUIRE EXITPROGRAM() NUMEXITS()
Program Status	Indicates whether this exit program is available for execution. Source field: EXEC CICS INQUIRE EXITPROGRAM() STARTSTATUS(cvda)
Program Concurrency	Indicates the concurrency attribute of this exit program. Source field: EXEC CICS INQUIRE PROGRAM() CONCURRENCY(cvda)

Table 301. Fields in the User Exit Programs report (continued)

Field Heading	Description
Exit Program Use Count	The number of times this exit program has been invoked. Source field: EXEC CICS INQUIRE PROGRAM() USECOUNT(data-area)
LIBRARY Name	The name of the LIBRARY from which the program was loaded. This is blank if the program has not been loaded, or if the LPASTATUS is LPA (indicating that the program was loaded from the LPA). Source field: EXEC CICS INQUIRE PROGRAM() LIBRARY(data-area)
LIBRARY Data Set Name	The name of the data set in the LIBRARY from which the program was loaded. This is blank if the program has not been loaded, or if the LPASTATUS is LPA (indicating that the program has been loaded from the LPA). Source field: EXEC CICS INQUIRE PROGRAM() LIBRARYDSN(data-area)
Program Name	The program name of the program that is enabled as an exit program by using the EXEC CICS ENABLE command. Source field: EXEC CICS INQUIRE EXITPROGRAM()
Entry Name	The entry point name for this exit program. Source field: EXEC CICS INQUIRE EXITPROGRAM() ENTRYNAME()
API	Indicates which APIs the task-related user exit program uses. The values are as follows: **CICSAPI** The task-related user exit program is enabled as either QUASIRENT or THREADSAFE, but without the OPENAPI option. The program is restricted to the CICS permitted programming interfaces. **OPENAPI** The task-related user exit program is enabled with the OPENAPI option. The program is permitted to use non-CICS APIs, for which purpose CICS will give control to the task-related user exit under an open TCB. OPENAPI assumes that the program is written to threadsafe standards. Source field: EXEC CICS INQUIRE EXITPROGRAM() APIST(cvda)

Table 301. Fields in the User Exit Programs report (continued)

<table>
<tr><th>Field Heading</th><th>Description</th></tr>
<tr><td>Concurrency Status</td><td>Indicates the concurrency attribute of the exit program. The values are as follows:

QUASIRENT
The task-related user exit program is defined as being quasi-reentrant, and can run only under the CICS QR TCB when invoking CICS services through the CICS API. To use any MVS services, this task-related user exit program must switch to a privately-managed TCB.

THREADSAFE
The task-related user exit program is defined as threadsafe, and can run under an open TCB. If the APIST option returns OPENAPI, the program is always invoked under an open TCB. If the APIST option returns CICSAPI, the program is invoked under the TCB that is in use by its user task when the program is given control, which might be either an open TCB, or the CICS QR TCB.

REQUIRED
The task-related user exit program is defined to always run on an open TCB. REQUIRED was specified either on the program definition or on the ENABLE PROGRAM command.
Note: When a task-related user exit is enabled REQUIRED and OPENAPI, it is treated the same as if it was enabled THREADSAFE and OPENAPI. For compatibility, an INQUIRE EXITPROGRAM for either combination always returns THREADSAFE, OPENAPI. For a task-related user exit enabled REQUIRED and CICSAPI, INQUIRE EXITPROGRAM returns REQUIRED, CICSAPI.

Source field: EXEC CICS INQUIRE EXITPROGRAM() CONCURRENST(cvda)</td></tr>
<tr><td>Qualifier</td><td>The name of the qualifier specified for this exit program.

Source field: EXEC CICS INQUIRE EXITPROGRAM() QUALIFIER()</td></tr>
<tr><td>Length</td><td>The length of the task local work area associated with this exit program.

Source field: EXEC CICS INQUIRE EXITPROGRAM() TALENGTH()</td></tr>
<tr><td>Task Related User Exit Options - Taskstart</td><td>Indicates whether this exit program was enabled with the TASKSTART option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() TASKSTART(cvda)</td></tr>
<tr><td>Task Related User Exit Options - EDF</td><td>Indicates whether this exit program was enabled with the FORMATEDF option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() FORMATEDFST(cvda)</td></tr>
<tr><td>Task Related User Exit Options - Shutdown</td><td>Indicates whether this exit program was enabled with the SHUTDOWN option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() SHUTDOWNST(cvda)</td></tr>
<tr><td>Task Related User Exit Options - Indoubt</td><td>Indicates whether this exit program was enabled with the INDOUBTWAIT option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() INDOUBTST(cvda)</td></tr>
<tr><td>Task Related User Exit Options - SPI</td><td>Indicates whether this exit program was enabled with the SPI option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() SPIST(cvda)</td></tr>
<tr><td>Task Related User Exit Options - Purgeable</td><td>Indicates whether this exit program was enabled with the PURGEABLE option.

Source field: EXEC CICS INQUIRE EXITPROGRAM() PURGEABLEST(cvda)</td></tr>
</table>

Virtual Hosts report

The Virtual Hosts report is produced using the **EXEC CICS INQUIRE HOST** command.

Table 302. Fields in the Virtual Hosts report

Field Heading	Description
Virtual Host name	The name of the virtual host. Source field: **EXEC CICS INQUIRE HOST**
TCPIPSERVICE name	The name of the TCPIPSERVICE definition that specifies the inbound port to which this virtual host relates. If this definition is not given, the virtual host relates to all TCPIPSERVICE definitions. Source field: **EXEC CICS INQUIRE HOST() TCPIPSERVICE**
Virtual Host Enable Status	Whether the virtual host is enabled or disabled, meaning that the URIMAP definitions which make up the virtual host can or cannot be accessed by applications. Source field: **EXEC CICS INQUIRE HOST() ENABLESTATUS**

Web Services report

The web services report is produced using a combination of **EXEC CICS INQUIRE WEBSERVICE** and **EXEC CICS EXTRACT STATISTICS WEBSERVICE RESID()** commands.

The statistics data is mapped by the DFHPIWDS DSECT.

Table 303. Fields in the WEBSERVICEs report

Field Heading	Description
WEBSERVICE Name	The name of the web service. Source field: **EXEC CICS INQUIRE WEBSERVICE**
WEBSERVICE Status	The state of the web service. Source field: **EXEC CICS INQUIRE WEBSERVICE() STATE**
Last modified date and time	The time, in milliseconds since 00:00 on January 1st 1900, that the deployed WSBind file on z/OS UNIX was last updated. Source field: **EXEC CICS INQUIRE WEBSERVICE() LASTMODTIME**
URIMAP name	The name of a dynamically installed URIMAP resource definition, if there is one that is associated with this web service. Source field: **EXEC CICS INQUIRE WEBSERVICE() URIMAP**
PIPELINE name	The name of the PIPELINE resource that contains this web service resource. Source field: **EXEC CICS INQUIRE WEBSERVICE() PIPELINE**
web service description (WSDL)	The file name of the web service description (WSDL) file associated with the web service resource. Source field: **EXEC CICS INQUIRE WEBSERVICE() WSDLFILE**
web service binding file	The file name of the web service binding file associated with the web service resource. Source field: **EXEC CICS INQUIRE WEBSERVICE() WSBIND**

Table 303. Fields in the WEBSERVICEs report (continued)

Field Heading	Description
web service WSDL binding	The WSDL binding represented by the web service. This binding is one of (potentially) many that appear in the WSDL file. Source field: **EXEC CICS INQUIRE WEBSERVICE() BINDING**
Endpoint	The URI specifying the location on the network (or endpoint) of the web service, as defined in the Web service description. Source field: **EXEC CICS INQUIRE WEBSERVICE() ENDPOINT**
Validation	Indicates whether full validation of SOAP messages against the corresponding schema in the web service description is specified. Source field: **EXEC CICS INQUIRE WEBSERVICE() VALIDATIONST**
Program interface	For a service provider, indicates whether CICS passes data to the target application program in a COMMAREA or a channel. Source field: **EXEC CICS INQUIRE WEBSERVICE() PGMINTERFACE**
Program name	The name of the target application program. Source field: **EXEC CICS INQUIRE WEBSERVICE() PROGRAM**
Container	When CICS passes data to the target application program in a channel, indicates the name of the container that holds the top level data. Source field: **EXEC CICS INQUIRE WEBSERVICE() CONTAINER**
WEBSERVICE use count	The number of times this web service was used to process a web service request. Source field: PIW-WEBSERVICE-USE-COUNT

WebSphere MQ Connection report

The WebSphere MQ Connection report is produced using the **EXEC CICS EXTRACT STATISTICS MQCONN** command. The statistics data is mapped by the **DFHMQGDS DSECT**.

Table 304. Fields in the WebSphere MQ Connection report

Field Heading	Description
MQCONN name	The name of the installed MQCONN definition for the CICS region, which defines the attributes of the connection between CICS and WebSphere MQ. Source field: MQG-MQCONN-NAME
WebSphere MQ Connection Status	The status of the connection between CICS and WebSphere MQ. Source field: MQG-CONNECTION-STATUS
WebSphere MQ connect date / time	The date and time when the most recent connection between CICS and WebSphere MQ was started. Source field: MQG-CONNECT-TIME-LOCAL
Mqname	The name of the WebSphere MQ queue manager or queue-sharing group that is specified in the MQNAME attribute of the installed MQCONN definition for the CICS region. CICS uses this as the default for the connection. Source field: MQG-MQNAME

Table 304. Fields in the WebSphere MQ Connection report (continued)

Field Heading	Description
WebSphere MQ Queue Manager Name	The name of the WebSphere MQ queue manager to which CICS is currently connected. If CICS is not connected to WebSphere MQ, this field is blank. Source field: MQG-QMGR-NAME
Resync group member	This shows whether the MQCONN definition for the CICS region specifies resynchronization if there are indoubt units of work when CICS reconnects to WebSphere MQ. Source field: MQG-RESYNCMEMBER
WebSphere MQ Release	The release of Websphere MQ that is connected to CICS. Source field: MQG-MQ-RELEASE
Initiation Queue Name	The name of the default initiation queue for the connection between CICS and WebSphere MQ. Source field: MQG-INITIATION-QUEUE
Number of current tasks	The number of current tasks that have issued an MQI call. Source field: MQG-TTasks
Number of futile attempts	A count of the number of MQI calls made while the connection status is "not connected". This is reset to zero when the connection is established. Source field: MQG-TFutileAtt
Total number of API calls	The total number of MQI calls since the connection was made. Source field: MQG-TApi
Number of API calls completed OK	The total number of calls that have completed successfully. Source field: MQG-TApiOk
API Crossing Exit Name	The name of the API-crossing exit, which is always CSQCAPX. Source field: not applicable
API Crossing Exit Concurrency Status	Whether the API-crossing exit is defined as QUASIRENT, THREADSAFE, or REQUIRED. Source field: EXEC CICS INQUIRE PROGRAM CONCURRENCY
Number of OPEN requests	The number of MQOPEN calls issued. Source field: MQG-TOPEN
Number of CLOSE requests	The number of MQCLOSE calls issued. Source field: MQG-TCLOSE
Number of GET requests	The number of MQGET calls issued. Source field: MQG-TGET
Number of GETWAIT requests	The number of MQGET calls issued with the MQGMO_WAIT option. Source field: MQG-TGETWAIT
Number of GETWAITs that waited	The number of MQGET calls issued with the MQGMO_WAIT option that waited for a message. Source field: MQG-TWaitMsg

Table 304. Fields in the WebSphere MQ Connection report (continued)

Field Heading	Description
Number of PUT requests	The number of MQPUT calls issued. Source field: MQG-TPUT
Number of PUT1 requests	The number of MQPUT1 calls issued. Source field: MQG-TPUT1
Number of INQ requests	The number of MQINQ calls issued. Source field: MQG-TINQ
Number of SET requests	The number of MQSET calls issued. Source field: MQG-TSET
Number of internal MQ calls	The number of internal MQ calls made. Source field: MQG-TCall
Number that completed synchronously	The total number of calls completed synchronously. Source field: MQG-TCallSyncComp
Number that needed I/O	The total number of calls that needed I/O. Source field: MQG-TCallIO
Number of calls with TCB switch	The number of API calls with a TCB switch. Source field: MQG-TSubtasked
Number of indoubt units of work	The number of indoubt UOWs at adapter startup. Source field: MQG-TIndoubtUOW
Number of unresolved units of work	The number of UOWs that were in doubt at adapter startup, and that have not been resolved because of a CICS cold start. Source field: MQG-TUnresolvedUOW
Number of resolved committed UOWs	The number of UOWs that were in doubt at adapter startup that have now been resolved by committing. Source field: MQG-TResolveComm
Number of resolved backout UOWs	The number of UOWs that were in doubt at adapter startup that have now been resolved by backing out. Source field: MQG-TResolveback
Number of Backout UOWs	The total number of backed out UOWs. Source field: MQG-TBackUOW
Number of Committed UOWs	The total number of committed UOWs. Source field: MQG-TCommUOW
Number of tasks	The total number of tasks. Source field: MQG-TTaskend
Number of Single Phase Commits	The total number of single-phase commits. Source field: MQG-TSPComm

Table 304. Fields in the WebSphere MQ Connection report (continued)

Field Heading	Description
Number of Two Phase Commits	The total number of two-phase commits. Source field: MQG-T2PComm
Number of CB requests	The number of MQCB calls issued. Source field: MQG-TCB
Number of msgs consumed	The number of messages passed to callback routines. Source field: MQG_TCONSUME
Number of CTL requests	The number of MQCTL calls issued. Source field: MQG-TCTL
Number of SUB requests	The number of MQSUB calls issued. Source field: MQG-TSUB
Number of SUBRQ requests	The number of MQSUBRQ calls issued. Source field: MQG-TSUBRQ
Number of STAT requests	The number of MQSTAT calls issued. Source field: MQG-TSTAT
Number of CRTMH requests	The number of MQCRTMH calls issued. Source field: MQG-TCRTMH
Number of DLTMH requests	The number of MQDLTMH calls issued. Source field: MQG-TDLTMH
Number of SETMP requests	The number of MQSETMP calls issued. Source field: MQG-TSETMP
Number of INQMP requests	The number of MQINQMP calls issued. Source field: MQG-TINQMP
Number of DLTMP requests	The number of MQDLTMP calls issued. Source field: MQG-TDLTMP
Number of MHBUF requests	The number of MQMHBUF calls issued. Source field: MQG-TMHBUF
Number of BUFMH requests	The number of MQBUFMH calls issued. Source field: MQG-TBUFMH

XMLTRANSFORMs report

The XMLTRANSFORMs report shows information and statistics about XMLTRANSFORM resources. The XMLTRANSFORM resource defines where the XML binding is located on z/OS UNIX and its status. CICS dynamically creates an XMLTRANSFORM resource when you install a BUNDLE or ATOMSERVICE resource.

This report is produced using a combination of **EXEC CICS INQUIRE XMLTRANSFORM** and **EXEC CICS EXTRACT STATISTICS** commands. The statistics data is mapped by the DFHMLRDS DSECT.

Table 305. Fields in the XMLTRANSFORMs report

Field Heading	Description
XMLTRANSFORM Name	The name of the XMLTRANSFORM resource definition. Source field: **EXEC CICS INQUIRE XMLTRANSFORM**
XMLTRANSFORM Enable Status	The status of the XMLTRANSFORM resource definition. Source field: **EXEC CICS INQUIRE XMLTRANSFORM () ENABLESTATUS**
XMLTRANSFORM XSDBIND File	The location of the xsdbind file in z/OS UNIX. Source field: **EXEC CICS INQUIRE XMLTRANSFORM () XSDBIND**
XMLTRANSFORM XML Schema File	The location of the XML schema file in z/OS UNIX. Source field: **EXEC CICS INQUIRE XMLTRANSFORM () XMLSCHEMA**
XMLTRANSFORM Msg Validation	The status of XML validation. Source field: **EXEC CICS INQUIRE XMLTRANSFORM () VALIDATIONST**
XMLTRANSFORM Use Count	The number of times that the xsdbind file has been used for data transformation. Source field: MLR-XMLTRNFM-USE-COUNT

Part 5. Appendixes

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who want to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Privacy Policy Considerations

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

CICSPlex SM Web User Interface :

For the WUI main interface: Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user's user name and other personally identifiable information for purposes of session management, authentication, enhanced user usability, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the WUI Data Interface: Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's user name and other personally identifiable information for purposes of session management, authentication, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the WUI Hello World page: Depending upon the configurations deployed, this Software Offering may use session cookies that collect no personally identifiable information. These cookies cannot be disabled.

For CICS Explorer: Depending upon the configurations deployed, this Software Offering may use session and persistent preferences that collect each user's user name and password, for purposes of session management, authentication, and single sign-on configuration. These preferences cannot be disabled, although storing a user's password on disk in encrypted form can only be enabled by the user's explicit action to check a check box during sign-on.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at http://www.ibm.com/privacy and IBM's Online Privacy Statement at http://www.ibm.com/privacy/details the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at http://www-01.ibm.com/software/info/product-privacy/.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Bibliography

CICS books for CICS Transaction Server for z/OS

General

CICS Transaction Server for z/OS Program Directory - base, GI13-3375
CICS Transaction Server for z/OS Program Directory activation module - base, GI13-3376
CICS Transaction Server for z/OS Program Directory activation module - Developer Trial, GI13-3377
CICS Transaction Server for z/OS Program Directory activation module - Value Unit Edition, GI13-3378
CICS Transaction Server for z/OS What's New, GC34-7437
CICS Transaction Server for z/OS Upgrading to CICS TS Version 5.3, GC34-7436
CICS Transaction Server for z/OS Installation Guide, GC34-7414

Access to CICS

CICS Internet Guide, SC34-7416

CICS Web Services Guide, SC34-7452

Administration

CICS System Definition Guide, SC34-7428
CICS Customization Guide, SC34-7404
CICS Resource Definition Guide, SC34-7425
CICS Operations and Utilities Guide, SC34-7420
CICS RACF Security Guide, SC34-7423
CICS Supplied Transactions, SC34-7427

Programming

CICS Application Programming Guide, SC34-7401
CICS Application Programming Reference, SC34-7402
CICS System Programming Reference, SC34-7429
CICS Front End Programming Interface User's Guide, SC34-7412
CICS C++ OO Class Libraries, SC34-7405
CICS Distributed Transaction Programming Guide, SC34-7410
CICS Business Transaction Services, SC34-7403
Java Applications in CICS, SC34-7417

Diagnosis

CICS Problem Determination Guide, GC34-7422
CICS Performance Guide, SC34-7421
CICS Messages and Codes Vol 1, GC34-7418
CICS Messages and Codes Vol 2, GC34-7419
CICS Diagnosis Reference, GC34-7409
CICS Recovery and Restart Guide, SC34-7424
CICS Data Areas, GC34-7406
CICS Trace Entries, SC34-7430
CICS Debugging Tools Interfaces Reference,GC34-7408

Communication

CICS Intercommunication Guide, SC34-7415
CICS External Interfaces Guide, SC34-7411

Databases

CICS DB2 Guide, SC34-7407

CICS IMS Database Control Guide, SC34-7413

CICS Shared Data Tables Guide, SC34-7426

CICSPlex SM books for CICS Transaction Server for z/OS

General

CICSPlex SM Concepts and Planning, SC34-7441
CICSPlex SM Web User Interface Guide, SC34-7451

Administration and Management

CICSPlex SM Administration, SC34-7438
CICSPlex SM Operations Views Reference, SC34-7447
CICSPlex SM Monitor Views Reference, SC34-7446
CICSPlex SM Managing Workloads, SC34-7444
CICSPlex SM Managing Resource Usage, SC34-7443
CICSPlex SM Managing Business Applications, SC34-7442

Programming

CICSPlex SM Application Programming Guide, SC34-7439
CICSPlex SM Application Programming Reference, SC34-7440

Diagnosis

CICSPlex SM Resource Tables Reference Vol 1, SC34-7449
CICSPlex SM Resource Tables Reference Vol 2, SC34-7450
CICSPlex SM Messages and Codes, GC34-7445
CICSPlex SM Problem Determination, GC34-7448

Other CICS publications

The following publications contain further information about CICS, but are not provided as part of CICS Transaction Server for z/OS, Version 5 Release 3 .

Designing and Programming CICS Applications, SR23-9692

CICS Application Migration Aid Guide, SC33-0768

CICS Family: API Structure, SC33-1007

CICS Family: Client/Server Programming, SC33-1435

CICS Family: Interproduct Communication, SC34-6853

CICS Family: Communicating from CICS on System/390, SC34-6854

CICS Transaction Gateway for z/OS Administration, SC34-5528

CICS Family: General Information, GC33-0155

CICS 4.1 Sample Applications Guide, SC33-1173

CICS/ESA 3.3 XRF Guide , SC33-0661

Other IBM publications

The following publications contain information about related IBM products.

CICS Performance Analyzer

CICS Performance Analyzer for z/OS User's Guide, SC34-2946
CICS Performance Analyzer for z/OS Report Reference, SC34-2947
CICS Performance Analyzer for z/OS Getting Started Guide, SC34-2945

DB2

DB2 Universal Database™ for OS/390 and z/OS Administration Guide, SC26-9931

DB2 Performance Expert for z/OS and DB2 Performance Monitor for z/OS

IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS: Report Command Reference, SC18-7977
IBM DB2 Performance Expert for z/OS; IBM DB2 Performance Monitor for z/OS: Report Reference,, SC18-7978

DFSMS

z/OS: DFSMSdfp Storage Administration Reference, SC26-7402

IMS

IMS: Administration Guide: Database Manager, SC26-8725
IMS: Administration Guide: System, SC27-1284
IBM IMS Performance Analyzer for z/OS: User's Guide, SC27-0912
IBM IMS Performance Analyzer for z/OS: Report Analysis, SC27-0913
IMS Database Tools Volume II: System Extension and Other Tools, SG24-5242

MVS

z/OS MVS Initialization and Tuning Guide, SA22-7591
z/OS MVS Initialization and Tuning Reference, SA22-7592
z/OS MVS JCL Reference, SA22-7597
z/OS MVS System Management Facilities (SMF), SA22-7630
z/OS MVS Planning: Global Resource Serialization, SA22-7600
z/OS MVS Planning: Workload Management, SA22-7602
z/OS MVS Setting Up a Sysplex, SA22-7625

z/OS Resource Measurement Facility (RMF)

Resource Measurement Facility User's Guide, SC33-7990
Resource Measurement Facility Performance Management Guide, SC33-7992
Resource Measurement Facility Report Analysis, SC33-7991
Resource Measurement Facility Programmer's Guide, SC33-7994

Language environment

z/OS Language Environment Programming Reference, SA22-7562
z/OS: Language Environment Concepts Guide, SA22-7567
z/OS: Language Environment Run-Time Migration Guide, GA22-7565
z/OS: Language Environment Programming Guide, SA22-7561
z/OS: Language Environment Customization , SA22-7564

Tivoli Decision Support for z/OS

Tivoli Decision Support for z/OS Administration Guide, SH19-6816
Tivoli Decision Support for z/OS CICS Performance Feature Guide and Reference, SH19-6820

NetView Performance Monitor (NPM)

NPM Reports and Record Formats, SH19-6965
NPM User's Guide, SH19-6962

Tuning tools

Network Program Products Planning, SC30-3351

Others

CICS Workload Management Using CICSPlex SM and the MVS/ESA Workload Manager, GG24-4286
System/390 MVS Parallel Sysplex Performance, GG24-4356
System/390 MVS/ESA Version 5 Workload Manager Performance Studies, SG24-4352
IBM 3704 and 3705 Control Program Generation and Utilities Guide, GC30-3008
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
Hierarchical File System Usage Guide, SG24-5482
A Performance Study of Web Access to CICS, SG24-5748

Accessibility

Accessibility features help users with a physical disability, for example restricted mobility or limited vision, to use information technology products successfully. CICS Transaction gateway is compatible with the JAWS screen reader. CICS Transaction GatewayCICS Universal Client provides accessibility by enabling keyboard-only operation.

For more information about the IBM commitment to accessibility, visit the IBM Accessibility Center.

Index

Numerics

D

E

M

N

O

Readers’ Comments — We'd Like to Hear from You

CICS Transaction Server for z/OS
Version 5 Release 3
Performance Guide

Publication No. SC34-7421-00

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your 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. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send a fax to the following number: +44 1962 816151
- Send your comments via email to: idrcf@uk.ibm.com

If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	Email address

Readers' Comments — We'd Like to Hear from You
SC34-7421-00

Cut or Fold
Along Line

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

PLACE
POSTAGE
STAMP
HERE

IBM United Kingdom Limited
User Technologies Department (MP189)
Hursley Park
Winchester
Hampshire
United Kingdom
SO21 2JN

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

SC34-7421-00

Cut or Fold
Along Line

SC34-7421-00