

English



BS2000/OSD

DAB V9.3

Disk Access Buffer

User Guide

Edition June 2012

Comments... Suggestions... Corrections...

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to:

manuals@ts.fujitsu.com

Certified documentation according to DIN EN ISO 9001:2008

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2008.

cognitas. Gesellschaft für Technik-Dokumentation mbH

www.cognitas.de

Copyright and Trademarks

Copyright © Fujitsu Technology Solutions GmbH 2012.

All rights reserved.

Delivery subject to availability; right of technical modifications reserved.

All hardware and software names used are trademarks of their respective manufacturers.

Contents

1	Preface	7
1.1	Objectives and target groups of this manual	8
1.2	Summary of contents	9
1.3	Changes since the last edition of the manual	11
1.4	Notational conventions	12
2	DAB caching	13
2.1	AutoDAB	13
2.2	Function of DAB within the HIPERFILE concept	14
2.2.1	ADM-PFA caching	15
2.2.2	(User) PFA caching	15
2.2.3	HIPERBATCH	17
2.3	Caching modes	19
3	Cache media	21
3.1	Main memory (MM)	22
3.2	Global storage (GS)	23
3.2.1	Structure of global storage	24
3.2.1.1	Physical storage organization	25
3.2.1.2	Logical storage organization	25
3.2.1.3	Local and global GS utilization	27
3.2.1.4	GS utilization under VM2000	28
3.2.2	Creating and releasing partitions	29
3.2.3	GS reconfiguration	30

4	DAB functions	33
4.1	Setting the cache parameters	33
4.1.1	Automatic caching (AutoDAB)	34
4.1.2	Manual assignment of files and caching parameters	36
4.2	DAB in multiprocessor operation	37
4.2.1	Caching for shared disks in the ADM-PFA concept	37
4.2.2	Caching for shared disks in the user PFA concept	39
4.3	Creating and releasing cache areas	44
4.3.1	ADM-PFA caching	44
4.3.2	PFA caching	45
4.3.3	Releasing cache areas during shutdown	46
4.4	Dynamic reconfiguration	47
4.4.1	Changing the configuration parameters of the cache area	47
4.4.2	Modifying the disk allocation or the entries in the file catalog	49
4.5	Restoring DAB cache areas in GS	51
4.5.1	Steps involved in cache area restoration	51
4.5.2	Restoring cache areas on servers with local GS utilization	52
4.5.3	Reconfiguration in the computer network (parallel HIPLEX)	53
4.5.3.1	Cache areas for exclusively allocated disks	54
4.5.3.2	PFA cache areas for shared imported pubsets	56

5	Notes on DAB usage and performance behavior	59
5.1	Efficient use of DAB	60
5.1.1	Read caching	60
5.1.2	Write caching	61
5.1.3	Read/write caching	62
5.1.4	Read caching of encrypted files	63
5.2	Notes concerning DAB usage	64
5.3	Using DAB in conjunction with other products	67
5.3.1	DAB and DRV	67
5.3.2	DAB and GSVOL	67
5.3.3	DAB and concurrent copy	68
5.3.4	DAB and external disk storage systems	68
5.3.5	DAB and SPACEOPT	69
5.4	Performance behavior	70
5.4.1	Improving I/O time	70
5.4.2	Reducing the I/O system load	73
5.4.3	CPU utilization with the various cache media	73
5.4.4	SM2 monitoring reports for DAB caching	74
5.4.5	Throughput optimization with AutoDAB (complex I/O load)	74
5.4.6	Throughput increase with DAB with an I/O load distributed on two servers in the XCS	77
6	Installing, starting and stopping DAB	81
6.1	Installing DAB	81
6.2	Starting and stopping the DAB subsystem	84

7	Commands	85
	FORCE-STOP-DAB-CACHING	
	Force release of existing ADM-PFA DAB cache area	86
	MODIFY-DAB-CACHING	
	Modify parameters of DAB cache area dynamically	89
	MODIFY-DAB-PARAMETERS	
	Modify the subsystem parameters dynamically	100
	SHOW-DAB-CACHING	
	Display information on current DAB configuration	102
	START-DAB-CACHING	
	Create ADM-PFA DAB cache areas	123
	STOP-DAB-CACHING	
	Release existing ADM-PFA DAB cache areas	139
8	Error handling	145
8.1	General information	145
8.2	Description of individual error situations	146
8.2.1	Failure of the connection between server and buffered disk	146
8.2.2	Disk error affecting the supported data areas	147
8.2.3	Detachment/failure of a GS unit with dual data storage	148
8.2.4	Cache memory errors with mono data storage	151
8.3	GS caching with DAB and disaster recovery	152
	Abbreviations and glossary	155
	Related publications	159
	Index	161

1 Preface

The DAB subsystem is the central BS2000/OSD cache handler for all CPU-adjacent cache media. CPU-adjacent means that no input/output peripherals (e.g. channel or IOP) are required for access to the cache medium. A cache is a buffer for frequently used data. Accessing this buffer considerably speeds up disk I/Os. Unlike physical disk I/Os, the buffer is accessed in parallel, i.e. there is no need for disk-specific serialization by the software/hardware.

DAB fulfils the following functions: configuring buffers (cache areas) in the cache media, providing information services for status inquiries, and handling I/Os with the cache media. After creating cache areas for files or disk areas, DAB is involved in all I/O operations for these files or disk areas in order to perform the desired buffering (caching).

The following are the main operating parameters of the cache areas managed by DAB:

- selection of the data areas to be served
- cache medium in which the buffer is to be created
- size of the buffer
- caching mode (read cache, write cache, read/write cache)
- size of the cache segments to be filled when data is cached
- caching method (displacement according to LRU or no displacement, i.e. resident buffering)
- with write caches, specification whether and when (i.e. at what cache occupancy level) write data in the cache is to be saved to disk.

“AutoDAB” includes functions for automatic and intelligent caching which carry out or facilitate correct selection of the above parameters. This greatly simplifies the administration of DAB cache areas while also improving cache memory utilization appreciably. If required, the operating parameters of such cache areas can also be set manually by systems support.

DAB functions can be controlled in two ways:

- using DAB commands (/START-DAB-CACHING, /MODIFY-DAB-CACHING, /MODIFY-DAB-PARAMETERS, /SHOW-DAB-CACHING and /STOP-DAB-CACHING)
- via the DMS interface (/MODIFY-PUBSET-CACHE-ATTRIBUTES and user-specific file attributes or additionally using /START- or STOP-FILE-CACHING)

Output in S variables is supported for the output of information relating to the configuration of the cache areas.

The openSM2 software product can be used during DAB operation to monitor how efficiently it is functioning and to initiate any tuning measures that may be required.

DAB V9.3 can run under BS2000/OSD-BC V9.0 and higher on S servers (/390 architecture) and on SQ servers (X86-64 architecture).

1.1 Objectives and target groups of this manual

This manual is intended for BS2000/OSD systems support staff. It describes the function and performance behavior of DAB as well as its user interfaces (commands).

1.2 Summary of contents

The present manual is divided into the following chapters:

Chapter 1: *Preface*

contains a brief description of the product DAB, target groups, a summary of contents and the changes made since the last version.

Chapter 2: *DAB caching*

provides information about automatic caching, on the function of DAB within the HIPERFILE concept and about caching modes and methods.

Chapter 3: *Cache media*

describes the cache media main memory (Main-Memory, MM) and particularly global storage (Global-Storage, GS), its structure and utilization concept.

Chapter 4: *DAB functions*

provides information about DAB caching techniques (particularly AutoDAB), DAB in multiprocessor operation, creating and releasing cache areas for ADM-PFA and PFA caching, dynamic reconfiguration changes during caching, and restoring DAB cache areas in GS.

Chapter 5: *Notes on DAB usage and performance behavior*

provides general information on using DAB (efficient use, behavior when used in conjunction with other products, performance behavior).

Chapter 6: *Installing, starting and stopping DAB*

deals with installing DAB and also describes how to start and stop DAB and what you need to look out for in this context.

Chapter 7: *Commands*

contains an overview of the DAB commands, followed by detailed descriptions of the individual commands.

Chapter 8: *Error handling*

provides information about how disk or cache errors are detected and handled.

A list of related publications and an index follow at the back of the manual.

Related publications are referred to in short form throughout the manual. The full title of each appears in the list at the back of the manual.

Readme file

The functional changes to the current product version and revisions to this manual are described in the product-specific Readme file.

Readme files are available to you online in addition to the product manuals under the various products at <http://manuals.ts.fujitsu.com>. You will also find the Readme files on the Softbook DVD.

Information under BS2000/OSD

When a Readme file exists for a product version, you will find the following file on the BS2000 system:

```
SYSRME.<product>.<version>.<lang>
```

This file contains brief information on the Readme file in English or German (<lang>=E/D). You can view this information on screen using the `/SHOW-FILE` command or an editor. The `/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=<product>` command shows the user ID under which the product's files are stored.

Additional product information

Current information, version and hardware dependencies, and instructions for installing and using a product version are contained in the associated Release Notice. These Release Notices are available online at <http://manuals.ts.fujitsu.com>.

1.3 Changes since the last edition of the manual

The following major changes have been made to the manual:

- /MODIFY-DAB-CACHING command,
new operand values AREA = *ADD-PUBSET(...) / *REMOVE-PUBSET(...) for adding a pubset to or removing a pubset from an automatic cache area (AutoDAB).
- /MODIFY-DAB-CACHING command,
new operand value CACHE-SIZE = *REDUCE(NEWSIZE=0) can be specified.
- /START-DAB-CACHING command,
new operand value AREA = *FILE(FILE-AREA=*NO), no file specification).
- /START-DAB-CACHING command,
new operand CACHE-MEDIUM = *MAIN-MEMORY(MEMORY=*STD / *ANY / *BELOW-MIN-MEM-SIZE / *ABOVE-MIN-MEM-SIZE) for defining the location of the cache area and its management data for systems in which dynamic main memory reconfiguration is possible.
- The /FORCE-DESTROY-CACHE and /SHOW-CACHE-CONFIGURATION commands are described in the “Commands” manual [4].
Their descriptions are no longer included in this manual.

The “Messages” chapter is no longer included.

You can find the messages on our manual server (URL: <http://manuals.ts.fujitsu.com>) using an HTML application and on the “BS2000/OSD SoftBooks” DVD instead of in the former “System Messages” manual.

1.4 Notational conventions

For the sake of simplicity and clarity, frequently used names are abbreviated as follows:

- **S servers** for the S series Business Servers)
- **SQ servers** for the SQ series Business Servers

In the examples the strings `<date>` and `<time>` specify the current outputs for date and time when the examples are otherwise independent of date and time.

The following typographical elements are used in this manual:



For notes on particularly important information

[]

References to other publications within the text are given in abbreviated form followed by numbers; the full titles are listed in the “References” section at the back of this manual.

`input`

Inputs and system outputs in examples are shown in typewriter font

2 DAB caching

This chapter provides information about automatic caching (AutoDAB), the function of DAB within the HIPERFILE concept (ADM-PFA caching and USER-PFA caching) and DAB caching modes (read cache, write cache and read/write cache).

2.1 AutoDAB

The most important consideration when using DAB is to decide which applications can be accelerated by DAB and which files or disks are to be cached with DAB. A good knowledge of the overall system I/O load and the I/O access behavior of performance-critical applications is required for optimum selection of the database for DAB use.

To facilitate the above, DAB offers automatic, intelligent selection of the data in addition to manual selection (through explicit specification of the files to be buffered). Automatic DAB mode can be set for each cache area; the base function is called AutoDAB. In the simplest case, the entire available cache memory is configured as a cache area by systems support for all disks that affect performance, and file selection is left to AutoDAB, which ensures that frequently used files are buffered with a good hit rate while files which react unfavorably to caching are not cached.

AutoDAB provides the following functions:

- files suitable for caching are selected automatically from a set of volumes by means of intelligent caching algorithms
- the prefetch factor is set to match the access profile of the selected files
- the cache utilization of the buffered files is monitored cyclically
- the AutoDAB functions are described in detail from [page 34](#).

The automatic caching functions are available under both the ADM-PFA and the user PFA interface (see next section). The cache areas are created with the `/START-DAB-CACHING AREA=*BY-SYSTEM` command (ADM-PFA caching) or the `/MODIFY-PUBSET-CACHE-ATTRIBUTES CACHED-FILES=*BY-SYSTEM` command (user PFA caching).

2.2 Function of DAB within the HIPERFILE concept

The BS2000/OSD HIPERFILE (**high-performance file** access) concept offers caching in various cache media both via the command interface of the subsystem involved (ADM-PFA caching) and via a uniform command interface integrated in DMS (user PFA caching). The following can be used as cache media: main memory (MM) and global storage (GS). ADM-PFA caching and user PFA caching are explained in more detail on pages 15.

i The global storage cache medium is available only on S servers.

The HIPERFILE concept is supported by the DAB subsystem. The DAB subsystem supports software caching in the cache media main memory and global storage. DAB performs the tasks of configuring and managing the cache areas.

No additional software is required to control hardware caching for external disk storage systems. The cache is managed automatically by the external disk storage system in this case.

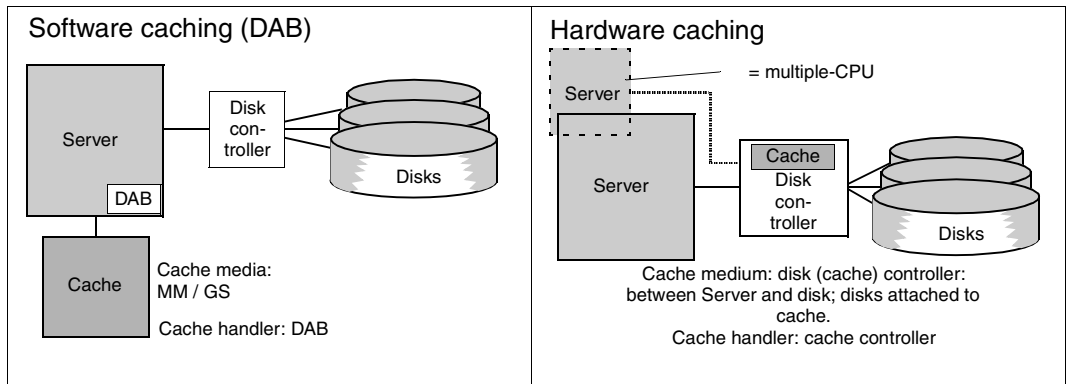


Figure 1: Comparison of software caching and hardware caching

2.2.1 ADM-PFA caching

With administrator-controlled (ADM-PFA) caching, systems support defines cache usage by means of DAB commands. Systems support decides which data media are to be supported by automatic caching (DAB selects the files) or which manually selected files or disks (entire or partial areas) would profit from caching. The data areas to be cached may be located on shared disks (public volumes) or private disks.

Systems support can create new ADM-PFA cache areas using the `/START-DAB-CACHING` command. ADM-PFA cache areas are independent cache units that are assigned, on the one hand, data areas to be served, and on the other hand, specific areas of cache storage. In addition, the following operating parameters are defined for each cache area:

- cache size, which implicitly defines the caching technique (displacement according to LRU or resident buffering)
- caching mode (read, write or read/write cache)
- size of cache segments (4, 8, 16 or 32 KB)
- saving of data to disk (controlled via a threshold value or not) for write and read/write caching
- location of the cache area and its management data
(resident below or non-resident above the minimum main memory size)

Files worth caching are selected automatically using AutoDAB. The cache segment size does not have to be selected since with automatic caching, DAB carries out the prefetch which matches the access profile of a file. With read/write caching, the data is also automatically saved to the disk.

2.2.2 (User) PFA caching

User PFA caching is described in full in the “Introductory Guide to Systems Support” [2]; only its basic features are therefore explained here.

User PFA caching will from here onwards be referred to in this manual as PFA caching.

PFA caching enables authorized users themselves to select which files are to be buffered. Alternatively, the files selected automatically by DAB or all files of a pubset can be buffered. The cached data is saved to the disk automatically when the files are closed, thus increasing security if cache errors occur.

The command interface for managing the pubset cache areas is integrated into DMS in the user PFA concept and is uniform for all cache media. Cache areas are created in two steps: cache area definition (`/MODIFY-PUBSET-CACHE-ATTRIBUTES` command) and its activation either when the pubset is imported or with the `/START-PUBSET-CACHING` command.

The definition of the PFA cache area for a pubset includes defining the cache medium and cache size. The settable cache definition parameters for the cache media served by DAB correspond on the whole to the settings for ADM-PFA caching which can be made with the `/START-DAB-CACHING` command.

Such a PFA cache area is released either automatically when the pubset is exported or with the `/STOP-PUBSET-CACHING` command.

PFA caching can be stopped explicitly for a file with the `/STOP-FILE-CACHING` command. The cache data is, if required, saved in the cache area and invalidated.

If the files to be cached are to be selected by the user (cache area defined with the command `/MODIFY-PUBSET-CACHE-ATTRIBUTES CACHED-FILES=*BY-USER-SELECTED`), systems support must explicitly grant the user ID appropriate authorization. User-specific caching attributes (PERFORMANCE, USAGE and DISK-WRITE) can then be set with either the `/CREATE-FILE` or the `/MODIFY-FILE-ATTRIBUTES` command.

For files which were already open before the setting up of the PFA cache area, BS2000/OSD V4.0 or higher allows PFA caching to be started later using the `/START-FILE-CACHING` command. The caching attributes (PERFORMANCE and USAGE) are specified directly in the command (in accordance with the user ID authorization).

As an alternative to the user assigning file cache attributes, systems support can also use the operand `CACHED-FILES=*ALL / *BY-SYSTEM` of the `/MODIFY-PUBSET-CACHE-ATTRIBUTES` command to define that all user files or the files of a pubset selected automatically by DAB are to be buffered. In this case, the dependencies of the operand `DISK-WRITE=*BY-CLOSE / *IMMEDIATE` (`/MODIFY-FILE-ATTRIBUTES` command) and `CACHE-MEDIUM` as well as for global storage with suboperand `VOLATILITY=*NO / *YES` (`/MODIFY-PUBSET-CACHE-ATTRIBUTES` command) must be noted, i.e. even with `CACHED-FILES=*ALL/*BY-SYSTEM` with the value `DISK-WRITE=*IMMEDIATE`, only read caching is employed in a volatile cache medium (`VOLATILITY=*YES`).

The home pubset cannot be buffered in any of the cache media managed by DAB with PFA caching. If there is write data in the cache area at the time of caching in the GS, this data cannot be reconstructed in the event of a system failure when the home pubset is imported, since the DAB subsystem is not yet available at this time. An attempt to set up a PFA cache for the home pubset is therefore rejected.

The two modes differ as shown below:

Characteristic	ADM-PFA caching	(User-)PFA caching
User interface	DAB commands	DMS commands, valid for all cache media
Selection of data to be buffered	Defined automatically or by systems support	Defined automatically by authorized user or all files in a pubset
Data for selecting the data to be buffered	Pubsets, volume sets, private disks or files	All files (with the exception of some system files) or files of a pubset selected by the user
Consistency points with write caching	Global consistency points, e.g. /STOP-DAB-CACHING or /EXPORT-PUBSET	When a file is closed, in the event of /STOP-FILE-CACHING for a file, DMS support

Table 1: Difference between ADM-PFA caching and user PFA caching

2.2.3 HIPERBATCH

The following section describes a means of speeding up batch processes with file follow-up processing steps, in order to combat continuously shrinking batch windows in the computer center.

HIPERBATCH (**high-performance batch** processing) designates the use of a special variant of PFA caching. Batch processing often comprises a sequence of processing steps for single files. For example, a (temporary) file is created in one step and this is reread as the input file and reused in a subsequent step. The file is generally closed and reopened between two such processing steps.

The following occurs after a file buffered with PFA is closed (if not otherwise explicitly specified):

- with a write cache, the data belonging to the file in the cache is saved to the disk (if required)
- the cache management data for this file is released by the cache handler DAB and any data belonging to the file which remains in the cache is invalidated.

This procedure increases data security in the event of a cache medium failure. However, for subsequent access, the file must first be reread into the cache before the application can profit from read hits, and the preceding write to disk is not mandatory during batch processing from the data security viewpoint since the run can be repeated if errors occur.

This is where HIPERBATCH comes in. A CLOSE parameter can be set to specify that when the file is closed, the data in the cache is not saved to the disk and particularly that the data is not invalidated. A subsequent OPEN to the same file can then immediately use the data in the cache. The effect is a noticeable speeding up of batch processes which include follow-up file processing. The CLOSE parameter can be specified via the /ADD-FILE-LINK command or the CLOSE program interface (see also “Commands” [4] and “Executive Macros” [1]):

```
/ADD-FILE-LINK    . . . ,CLOSE-MODE=*KEEP-DATA-IN-CACHE
```

or

```
CLOSE            <fcb> ,KEEP-DATA-IN-CACHE
```

If the /STOP-FILE-CACHING command is specified for a file that was closed in this CLOSE mode, the data in the cache is saved and invalidated (except in a pure read cache).



This procedure is generally used with ADM-PFA caching in order to achieve optimum performance. No additional parameters need to be specified for processing.

2.3 Caching modes

There are three basic caching modes: read cache, write cache and read/write cache. In the diagrams below, read accesses are represented by arrows on the left and write accesses by arrows on the right.

In the following section, a cache operated in the mode “read caching”, “write caching” or “read/write caching” is also referred to as read cache, write cache or read/write cache.



The DAB functions are executed under the user task. A separate system task is only created for asynchronous saving to disk.

Read caching

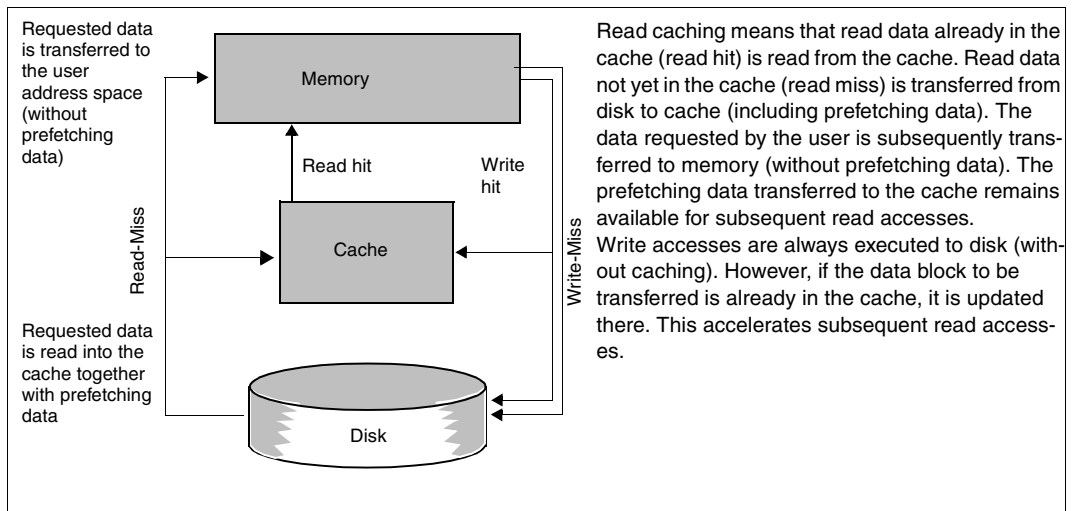


Figure 2: I/O sequence with read caching

Write caching

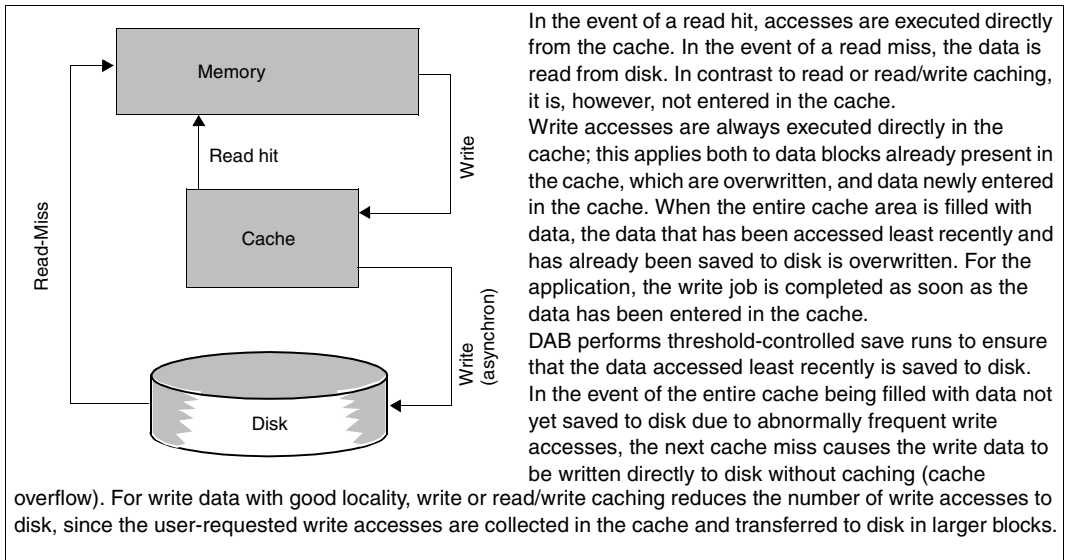


Figure 3: I/O sequence with write caching

Read/write caching

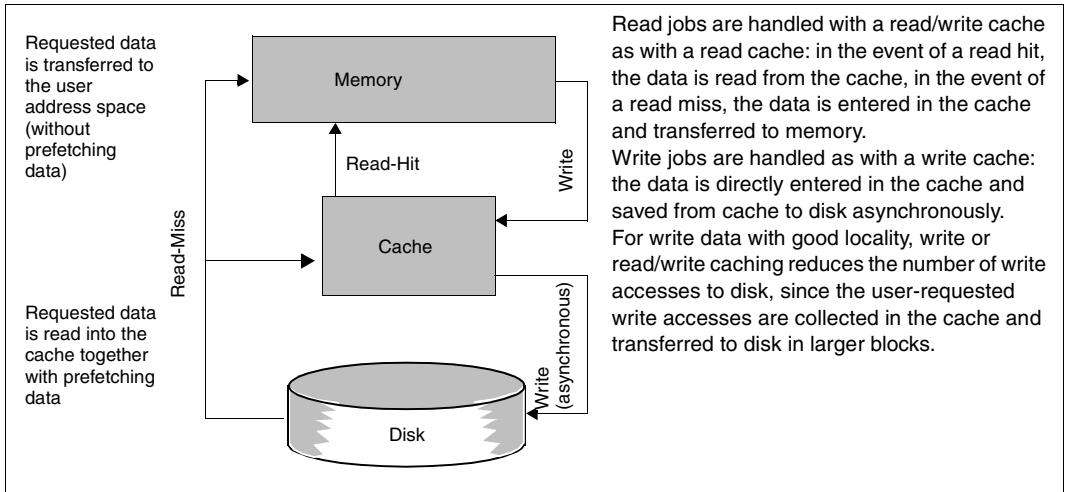


Figure 4: I/O sequence with read/write caching

3 Cache media

This chapter provides information about the storage media supported by DAB:

- main memory (MM)
- global storage (GS)

All these storage media have the following common features:

- fast access times
- no significant wait times between the start and end of a data transfer and consequently no system loading from task changing in this phase
- no system loading from interrupt handling at the end of a data transfer

The administration data required by DAB in the main memory resides in resident data spaces of the Enterprise System Architecture (ESA). The number of these data spaces is limited to 1024 by a hardware restriction, but because of other users (components of the operating system) fewer data spaces may be available. Depending on the cache medium, this will determine the maximum number of DAB cache areas.

3.1 Main memory (MM)

The DAB subsystem permits main memory areas of any BS2000/OSD system to be used for caching.

Main memory is best suited as a read cache since it is a volatile cache medium and data stored in it is no longer accessible after a system crash. If it is to be used as a write or read/write cache, this type of use is recommended for the following:

- files requiring no higher failsafe level during processing (temporary work files, SYSEAM)
- files whose write data is protected by an additional mechanism (e.g. logging)
- files that can be fully restored without excessive effort after a system crash.

Main memory cache areas can exceed 2 GB; their maximum size is currently only restricted by the physical memory available.

DAB requires at least 2 data spaces (for cache administration data) per cache area in the main memory, the maximum number of main memory cache areas is therefore 512 (if there are no other users of resident data spaces in the system).

Information on performance is provided in [section “CPU utilization with the various cache media” on page 73](#).

3.2 Global storage (GS)

Global storage, a semiconductor-based storage expansion, can be connected to S servers. Global storage is not supported on SQ servers.

Global storage comprises a maximum of two mutually independent hardware units, the so-called GS units. Each GS unit has its own memory controller and power supply. The maximum size of the GS and the number of GS units depends on the type of server it is connected to.

GS can be shared by several servers of the same type. In some cases the GS can also be used by servers of different types (in accordance with the information in the "Introductory Guide to System Support" [2]). A hardware network of this type is known as a parallel HIPLEX. The maximum number of servers in a parallel HIPLEX is 4, depending on the type.

The particular advantages of caching in GS are as follows:

- The data in GS is available when a system is restarted after a crash, with the same data state as before the crash
- In GS comprising two GS units, data can be stored using the dual recording method, i.e. the same data is transferred simultaneously to both units with one transfer operation (see also [section "DAB and DRV" on page 67](#)). This considerably increases data availability since the data can only be lost if errors occur simultaneously in the relevant storage areas of both units.
- GS can optionally be equipped with a backup battery which can maintain the power supply when emergencies occur.

GS therefore offers unrestricted write caching as well as the highest possible data availability.

Global storage is managed via the GSMAN subsystem (serving as controller and monitor). GSMAN must be installed as a subsystem on all servers with GS access. Subject to this condition, GSMAN is loaded automatically on all these servers at BS2000 initialization.

3.2.1 Structure of global storage

Utilization of global storage is organized in various levels:
GS unit → partition → cache area.

While the GSMAN subsystem subdivides the GS into partitions (administrative units of GS), the GS users (DAB with subdivision into cache areas, GSMAN, GSVOL etc.) subdivide the partitions - see the “Introductory Guide to Systems Support” [2]). Partition names have to comply with the naming conventions of the subsystems that use them.

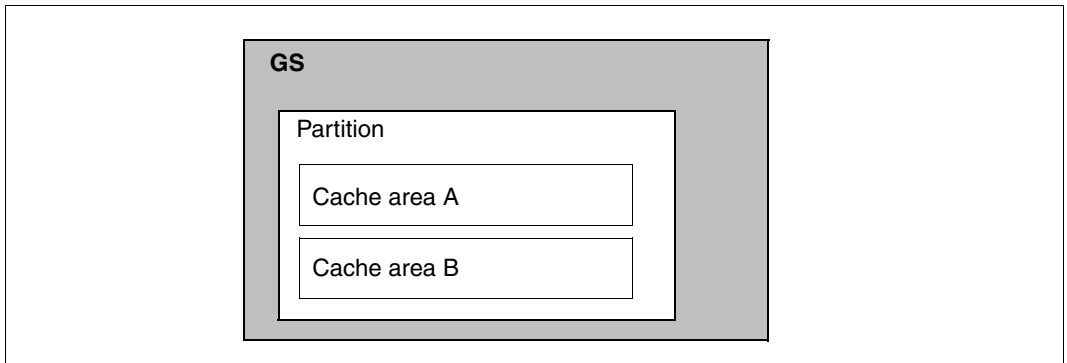


Figure 5: Levels of GS utilization

Partitions can be created as mono or dual partitions:

Mono partitions consist of a contiguous address space located on a single GS unit, while dual partitions consist of two contiguous address spaces located on two different GS units. All GS users can simultaneously use multiple mono partitions and/or dual partitions. Systems support is responsible for subdividing the GS into partitions by means of specific GS commands (see overview of commands on [page 29](#)). When planning GS utilization, it is advisable to create dual partitions first, prior to creating mono partitions.

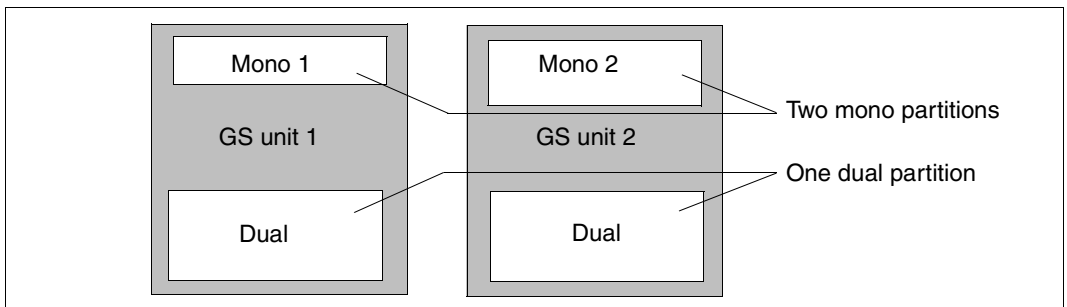


Figure 6: Mono and dual partitions

3.2.1.1 Physical storage organization

As mentioned above, global storage can consist of one or two mutually independent GS units. Since each GS unit is an autonomous device, hardware faults will normally affect one GS unit only.

Configuration options that offer optimum data security and storage utilization are made possible by the fact that the user interfaces of GSMAN, DAB and PFA permit explicit specification of the GS units on which the user-defined logical storage units, partitions and cache areas are to be located.

3.2.1.2 Logical storage organization

To create a DAB cache area in global storage, a DAB-GS partition must be available in which the new cache area can be embedded. GS partitions are created and released using GSMAN commands (/CREATE-GS-PARTITION and /DELETE-GS-PARTITION, in exceptional cases /FORCE-DESTROY-GS-PARTITION).

How many and which GS partitions have to be defined for caching with DAB depends on the configuration of the actual GS and that of the database.

The identifier for a DAB-GS partition must always have the following structure:

PARTITION-ID=DAB<x><catid>

where

<x>: Locality ID/mode of the partition.
The following values are possible:

- 1: The partition to be used for mono recording is located on GS unit 1.
- 2: The partition to be used for mono recording is located on GS unit 2.
- D: The partition is to be used for dual recording of cache data.

This part of the partition identifier corresponds to the GS-UNIT parameter in the cache area definition, i.e. a new cache area can only be embedded in a GS partition whose locality identifier is identical to that of the specification of the GS unit attribute of the cache area concerned.

<catid>: Catalog ID (Catid)

A new cache area is only created in a GS partition which corresponds to its pubset reference. The table below shows how the pubset reference of a cache area is determined:

Caching method	Caching mode ¹⁾	Special database attributes	Pubset reference
ADM-PFA	Read cache	x	<home>
ADM-PFA	Write cache	Data on (several) SF pubsets or (several) private disks	<home>
ADM-PFA	Write cache	Data of an SM pubset	<catid>
User PFA	x	Data of an exclusively imported SF pubset	<cat-id> / <home>
User PFA	x	Data of an SM pubset	<catid>
User PFA	x	Data of a shared pubset (SF or SM pubset)	<catid>

Table 2: Pubset reference of a cache area

Notes on the table:

x: this attribute is irrelevant in this context or not defined

1) read/write cache is handled in the same way as a write cache

<home> pubset ID of the actual home pubset of the system

<catid> pubset ID (only meaningful if the data of only one pubset is to be buffered)

As soon as the user defines a DAB GS partition, the GSMAN subsystem reserves one contiguous GS area per GS unit for this partition.

3.2.1.3 Local and global GS utilization

The availability of global storage for DAB depends on how GS is utilized by the server:

- Local GS utilization:
GS is used locally (i.e. exclusively) by a system if the GS hardware is only accessible on this system or if GSUSAGE = LOCAL is specified in the GSMAN parameter file. In such a configuration, GS is available to DAB as soon as the GSMAN subsystem has been loaded, i.e. prior to “System Ready”.
- Global GS utilization in a parallel HIPLEX:
A system participates in global GS utilization in a parallel HIPLEX (i.e. shares the GS with other systems) if GSUSAGE = GLOBAL is specified in the GSMAN parameter file and it is part of an XCS network. In such a configuration, GS is not available to DAB until the systems involved have been interconnected in the XCS network that is indispensable for global storage utilization, i.e. only after “XCS Ready”.

The assignment of cache areas or the data areas served by them to GS partitions is irrelevant in a local GS configuration (no parallel HIPLEX).

However, the situation changes with a global GS configuration in which two or more systems use the same GS. A separation of the GS slices for the individual users and systems is required here; this is achieved by creating GS partitions. GS partitions can also be shared by multiple systems (e.g. for PFA cache areas on shared pubsets).

A global GS configuration is also required in order to transfer cache areas to other servers in the network in the event of a failure (e.g. system crash). In this case, it is not possible to reassign individual cache areas from one system to another; the entire GS partition of the cache areas has to be reassigned.

A GS partition forms a **reattachment unit** in the parallel HIPLEX. A partition should contain cache areas whose entire data areas (disks) can be transferred to the same servers in the parallel HIPLEX.

With PFA cache areas, the reattachment unit is restricted to exactly one pubset (exception: a separate GS partition is not created for an exclusively imported pubset).

With ADM-PFA cache areas, the reattachment unit consists of all cached SF pubsets and private disks.



- In a parallel HIPLEX it should be possible to reassign all cached disks to all servers in the HIPLEX in order to avoid complications in the event of possible transfers of cache areas (and consequently of GS partitions).
- GS cache areas configured using a version of DAB other than that currently installed cannot be reconstructed. Before changing the DAB version, all DAB cache areas in global storage must be detached and the DAB GS partitions released.

3.2.1.4 GS utilization under VM2000

Three variants are possible when utilizing GS under VM2000 (see also the manual “VM2000” [8]):

- The entire body of real GS is assigned to a VM that uses it exclusively.
- The entire body of real GS is assigned to multiple VMs that share it.
- The entire body of real GS is shared among multiple VMs as virtual GS. In this case, the entire body of real GS is assigned to the monitor VM and multiple GS partitions with reserved names are generated. Each of these GS partitions can be assigned to a VM as virtual GS using VM2000 commands for either exclusive or shared utilization.

The GS utilization mode to be used depends on the backup concept and must be clarified before GS is used (if necessary in conjunction with consultants from the Competence Center).

Utilization of GS under VM2000 imposes constraints in terms of local or global utilization (see the section [“Local and global GS utilization” on page 27](#)).

Global utilization is possible in the following cases:

- GS is used on a server as virtual GS and the corresponding GS partition is assigned to multiple VMs on this server for shared utilization.
- GS is used on a server as real GS and is assigned to multiple VMs on this server for shared utilization.
- GS is used on a server as real GS and is exclusively assigned to a VM on this server. This VM shares the GS with another server (or a VM running on it) in the GS network.

In contrast, local utilization is required in the following cases:

- GS is used on a server as virtual GS and the corresponding GS partition is exclusively assigned to a VM on this server.
- GS is used on a server as real GS and is exclusively assigned to a VM on this server. GS cannot be accessed by any other server.

3.2.2 Creating and releasing partitions

The GS partitions must be defined by systems support before DAB can use the global storage.

GS partitions are created by means of the `/CREATE-GS-PARTITION` command and occupied by DAB when the first cache area is created in this partition. The size of a partition can only be modified by releasing and recreating a GS partition. All cache areas using a GS partition must be released (by means of DAB or PFA commands) prior to releasing the partition. A GS partition is released by DAB when its last cache area is released.

Refer to the “Introductory Guide to Systems Support” [2] for detailed information about creating and releasing GS partitions.

Commands for GS operation

Command	Function
ATTACH-GS-UNIT	Re-enables a GS unit (only possible if preceded by a <code>/DETACH-GS-UNIT</code> command).
CREATE-GS-PARTITION	Creates GS partitions, defines configuration data for a partition and physically sets up the defined partition.
DELETE-GS-PARTITION	Deletes the configuration data for a partition.
DETACH-GS-UNIT	Disables a GS unit.
FORCE-DESTROY-GS-PARTITION	Forcibly releases (destroys) a partition in GS.
SHOW-GS-STATUS	Retrieves the current GS configuration and the setting for the <code>GSUSAGE</code> subsystem parameter.

Table 3: Overview of commands for GS operation

Example of a GS configuration

The GS partitions are created once GSMAN is loaded. These commands are best stored in an ENTER file by systems support although reconfiguration will not be necessary unless both GS units are erased, e.g. as a result of the SVP function `RESET-SSU` or after `POWER-OFF`.

```
/CREATE-GS-PARTITION PARTITION-ID=DABDCAT1,SIZE=64,MODE=*DUAL _____ (1)
/CREATE-GS-PARTITION PARTITION-ID=DAB1CAT2,SIZE=64,MODE=*MONO(GS-UNIT=1) (2)
/CREATE-GS-PARTITION PARTITION-ID=DAB2CAT3,SIZE=32,MODE=*MONO(GS-UNIT=2) (3)
```

DAB partitions must be created in GS to permit GS utilization with DAB (DAB partition names must start with “DAB”):

- (1) A DAB dual partition is created.
- (2) A DAB mono partition is created on GS unit 1.
- (3) A DAB mono partition is created on GS unit 2.

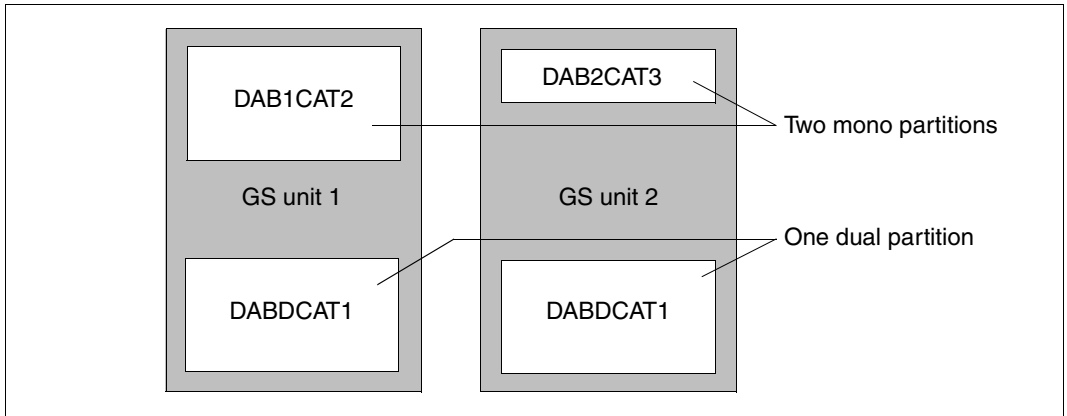


Figure 7: Example of a GS configuration

3.2.3 GS reconfiguration

GS reconfiguration refers to the attachment and detachment of GS units using the hardware interfaces provided. GS units are detached explicitly using the `/DETACH-GS-UNIT` command and implicitly in the event of a GS unit failure. GS units are attached explicitly using the `/ATTACH-GS-UNIT` command and implicitly when the system starts.

Detaching a GS unit

GS units can be shut down using the `/DETACH-GS-UNIT` command for maintenance work on the power supply, for example.

The `/DETACH-GS-UNIT GS-UNIT=x` command will only be accepted if there is no mono partition in use on the GS unit in question, i.e. no cache area with `GS-UNIT=1/2`. Additionally, no dual partition may be in use if the GS unit being detached is the last one. If a GS unit fails when the system is active the system will detach the relevant GS unit (message EGC2100), regardless of the current utilization modes.

Dual partitions on the remaining GS unit can still be accessed once a GS unit has been detached. Note, however, that the detached GS unit will contain obsolete data once the remaining GS unit is accessed for writing.

From the point of view of DAB, both the explicit detachment of a GS unit and implicit detachment in the event of the failure of a GS unit will result in access errors when writing to GS. DAB performs error handling on the foot of these access errors. The type of error handling performed will differ depending on whether the error is on a dual partition or a mono partition. The procedure to be adopted for errors on a dual partition can be set via parameters in the subsystem initialization file (see the parameters on [page 82](#) and, above all, the [section “Detachment/failure of a GS unit with dual data storage” on page 148](#)).

Attaching a GS unit

DAB permits the attachment of a second GS unit while the system is active, i.e. not all cache areas with GS-UNIT=*DUAL have to be detached before a GS unit can be attached. The attachment of GS units in both local GS and global GS in the XCS network is supported.

The `/ATTACH-GS-UNIT` command automatically equalizes all dual partitions containing valid data, i.e. the data on such partitions is copied from the remaining GS unit to the GS unit to be attached. This command may therefore take several minutes to execute. During the equalization process, DAB synchronizes write access operations on the GS units with the equalization process. The dual partitions will be identical following the successful execution of the command with valid data on both GS units.

If the first GS unit is reattached following the breakdown of the connection between GS and the server (i.e. following the total failure of GS), DAB checks whether the GS unit contains data that is still valid. If not, all GS access will continue to be suppressed (results in IO errors) to prevent the production of inconsistent data.

4 DAB functions

This chapter provides information about the following:

- setting the cache parameters (automatic caching, explicit setting, size of the cache area, special features with file encryption)
- creation and release of (ADM-PFA and PFA) cache areas
- DAB caching techniques
- dynamic reconfiguration
- reconstruction of DAB cache areas in GS.

4.1 Setting the cache parameters

The cache hit rate, i.e. the ratio of read and write hits to the total number of accesses, is decisive for system performance. Apart from the available cache size, the hit rate is dependent on the probability of the data last accessed or adjacent data being processed again, i.e. on the spatio-temporal locality of accesses. The cache hit rate depends not only on the selection of the data (files) to be cached, but also on the setting selected for the cache parameters.

DAB offers the following two options for deciding on which applications are to be expedited by DAB and which files or volumes should profit from DAB caching:

- Automatic caching (AutoDAB)
Here DAB selects the files to be cached itself on the basis of its observation of the data flow and cache hit behavior, and adjusts the settings dynamically. This ensures that only files with a good cache hit rate are buffered.
- Manual caching
Here users select the files to be cached themselves on the basis of their detailed knowledge of applications and their data access behavior, and also choose corresponding cache parameters. They must handle dynamic adjustments themselves.

4.1.1 Automatic caching (AutoDAB)

Automatic caching greatly relieves systems support of its previous cache administration tasks while also providing optimum use of the available cache.

AutoDAB carries out the tasks described below.

Automatic selection of suitable files on a specified set of data media

Enters the files into the cache configuration when they are opened. Files already open when a cache area is created are entered during the first I/O to the cache area.



Files added to or removed from the database with the `/MODIFY-DAB-CACHING` command cannot be added to or removed from the current cached file configuration by the file controller of AutoDAB.

File classification according to their access profile

The files are allocated to one of the following three classes according to their access profile:

- sequentially processed file
- non-sequentially (random) processed file with temporal locality
- non-sequentially (random) processed file without temporal locality

Caching according to analysis results

- Sequentially processed files are well suited for caching if a large prefetch is executed during reading and saving to the disk is executed in large blocks. During reading with AutoDAB, several small cache segments are therefore filled with one cache I/O (multi-segment prefetch) and when saving with AutoDAB, several small segments are written with adjacent data with a single cache I/O (multi-segment destage).

In addition, asynchronous caching is initiated with read accesses in case the data areas of the next prefetch I/O required have not yet been written into the cache. This further reduces the wait time for writing to the cache although in an ideal case the data is already in the cache before it is accessed.

If the space allocated to the cache area is already fully occupied and the supported data area exceeds the size defined in the subsystem initialization file, the system minimizes the occupation of cache segments for caching files of this class. This allows a data segment which is occupied for a data prefetch (from the disk) because of a read miss to be released for further data caching once all PAM pages in the segment have been referenced.

- Non-sequentially (random) processed files with temporal locality are well suited for caching if, as far as possible, only the referenced data (which will very probably be rereferenced soon because of the temporal locality) is written to the cache to improve utilization of the cache area. AutoDAB takes this into consideration when caching data and executes a minimum prefetch (record level caching).
- Non-sequentially (random) processed files without temporal locality are hardly suitable for caching since they cause continuous cache writes (or asynchronous saves) without rereferencing the data. This displaces other files, which utilize the cache better, from the cache. For this reason, AutoDAB excludes these files from caching once it detects this access behavior (after a synchronization phase). This class of files is only buffered if they fit completely into the cache without displacing other files.

Cyclic monitoring of the cache utilization of buffered files

The selected files (also those which are accessed non-sequentially, i.e. randomly) are initially buffered for a while before a check is made as to whether it is a good idea (as a whole) to cache the file. This means that checks are made at regular intervals to determine whether the performance of the files supported for the cache area is (still) satisfactory. The actual cache utilization rate is used as the criterion for this. The cache utilization rate determines the relationship between executed accesses and the segment allocations required for them. A large cache utilization rate value also means good cache utilization.

If the performance is poor for the supported files, caching is stopped for files with poor cache utilization.

Automatic FORCE-OUT correction

When a cache area is defined, systems support can use the command `/START-DAB-CACHING` with the operand `FORCE-OUT=...` to specify the threshold as of which write data is saved asynchronously from the cache to the relevant data storage media. This also defines which part of a cache area is to be held free for read data or caching new data, as far as possible. If this value is incorrectly set, it can have severe adverse effects on the performance of a cache area in borderline cases. This setting is therefore checked during automatic caching and corrected if necessary.

The possible values for `FORCE-OUT=...` are arranged in the following order:

1. `*NO`
2. `*AT-HIGH-FILLING`
3. `*AT-LOW-FILLING`

The value specified for an operand by the user is corrected if processing the allocation of a new cache segment causes a cache overflow. The system then switches to the next lower level if possible. If no further switching is possible and cache overflows occur during several monitoring periods, systems support is informed via a console message that the cache area is defined too small for the current load.

Caching data areas on shared private disks or shared pubsets

Shared allocated disks are served by AutoDAB with read caching. In contrast to the previous DAB ADM-PFA support, it is no longer necessary to ensure that disks are only read. DAB guarantees the data consistency of read caching by only reading files used on the local system (processed either without shared update or only with local shared update) and invalidating the cache contents when the files are closed.

This ensures that the latest data is accessed after the files have been updated from other systems. Files opened on shared pubsets in a HIPLEX in shared update mode and with `LOCK-ENVIRONMENT=*XCS`, i.e. which are subject to a system-global shared update, are not served by DAB in this case.

4.1.2 Manual assignment of files and caching parameters

The `/START-DAB-CACHING` command with the specification `AREA=*FILE(...)` enables the ADM-PFA data areas to be selected manually at file level. Manual file selection is set for user PFA by means of `/MODIFY-PUBSET-CACHE-ATTRIBUTES` with the specification `CACHED-FILES=*BY-USER`. Empirical values for the hit rate should be taken into consideration when selecting the files.

If you later wish to modify the data for a file-specific cache area, you can do this using the `/MODIFY-DAB-CACHING` command with the specification `AREA=*ADD-FILE(...)` or `*REMOVE-FILE(...)` (see the [section “Changing the configuration parameters of the cache area”](#) on page 47).

4.2 DAB in multiprocessor operation

Shared pubsets and shared private disks can be supported by all cache media (main memory and global storage) available in the HIPERFILE concept. The DAB caches are managed by the DAB instances of the individual systems.

When it comes to the options available for DAB caching of data areas on shared disks, i.e. to shared pubsets or shared private disks, the possible caching variants must be looked at individually. The following table provides a basic summary of the executions:

	ADM-PFA (public and private disks)	USER-PFA (pubsets)
Read caching	AutoDAB: system monitoring possible Manual DAB: within the responsibility of systems support	Read caching is the default for volatile cache media. Only locally processed files profit here.
(Read/)write caching	not supported	Read/write caching is the default for non-volatile cache media. Only locally processed files profit here.

Table 4: DAB caching of shared allocated disks

In the following sections, the options for DAB caching in multiprocessor operation for the ADM-PFA concept and the function “Read/write caching for shared pubsets in the user PFA concept” are described.

4.2.1 Caching for shared disks in the ADM-PFA concept

Read caching with ADM-PFA commands with AREA=*FILE (i.e. without automatic caching)

Caching data areas of this type with DAB is critical unless all systems involved only read the data. Otherwise, such data areas should not be served by DAB, not even in read caching mode.

The SHARED-DISK-SUPPORT operand of the /START-DAB-CACHING command can be used to define whether such shared data areas are to be buffered or not (*NO is the default).

When allocating a disk, DAB adjusts its current allocation state of the cache areas (SHARED/EXCLUSIV) with the SHARED-DISK-SUPPORT attribute. Areas of a shared volume are only served if the corresponding cache area was created with SHARED-DISK-SUPPORT=*YES. The F5 and F1 labels are never served.

Read caching with ADM-PFA commands with AREA=*BY-SYSTEM

Data consistency when read caching shared volumes is guaranteed by DAB. Systems support does not have to check to ensure that the served files are only read by all systems.

In this case, files that are only being used locally at the time of processing are cached. Data consistency is ensured by DMS on the one hand since it prevents the files from being opened by another computer during processing. On the other hand, DAB also ensures data consistency with the following measures:

When a file is closed, all copies of it in the cache are invalidated so that any subsequent (read) accesses to this file must always be to the disk. This ensures that the latest data is always processed, even after an (intermediate) data update by another system.

The SHARED-DISK-SUPPORT operand is meaningless in this case and is ignored.

(Read/write) caching with ADM-PFA commands

In addition to the explicit setting of write or read/write caching, it is also possible to set read/write caching using CACHING-MODE=*BY-CACHE-MEDIUM, depending on the cache medium and the data area specification.

Caching shared data areas in write or read/write mode is not possible and is prevented by DAB. The SHARED-DISK-SUPPORT operand is not processed in this case and an appropriate message is output.

Data areas on shared private disks or shared pubsets are not served in these modes.

4.2.2 Caching for shared disks in the user PFA concept

DAB supports the caching of the data areas of a shared pubset for a “homogeneous” CCS network (see the “HIPLEX MSCF” [3] manual). “Homogeneous” means that a BS2000/OSD version as of V8.0 is running on all systems that have access to the shared pubset.

The cache areas on the relevant pubset sharers may be located on a local medium (main memory or local GS) or in a globally used GS (i.e. there is a parallel HIPLEX).

The cache areas are configured by means of the MRSCAT entry for the shared pubset (see the `/MODIFY-PUBSET-CACHE-ATTRIBUTES` command). With SF pubsets, the cache parameters can be set separately for each pubset sharer; with SM pubsets the settings are valid for all participants in the shared pubset network. These cache areas are created during import to the relevant pubset sharer and are released again during export. The cache medium used must be homogeneous in the shared pubset network, i.e. it is not possible to cache with main memory on one pubset sharer and with (local) GS on another. This is checked by the system during creation of the cache areas. Also valid are homogeneity conditions concerning the BS2000/OSD versions (as of V8.0) participating in the shared pubset network, concerning the GS unit, and concerning GS utilization (local/global) for GS cache areas.

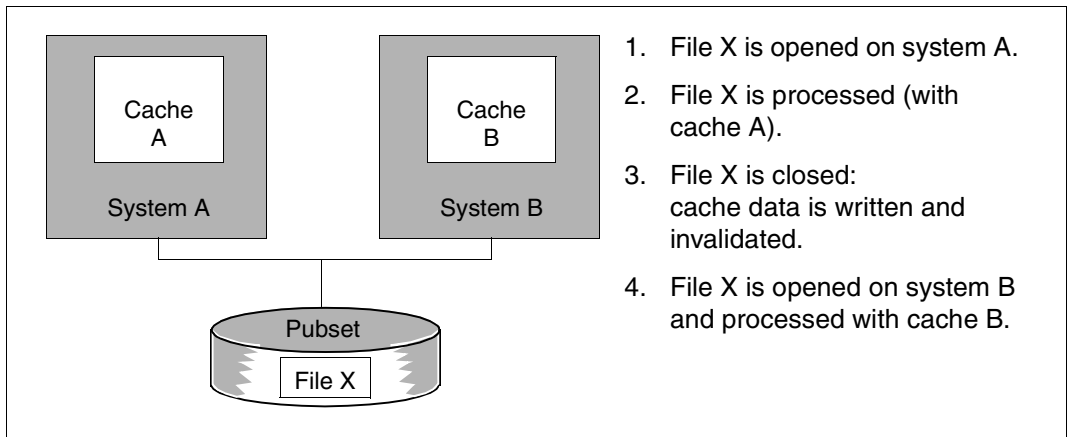


Figure 8: Concept for supporting shared pubsets with DAB

The procedures for the individual media are described in more detail below:

- caching in main memory
- caching in global storage, with a distinction being made between
 - caching in the local GS
 - caching in the global GS

Caching in main memory

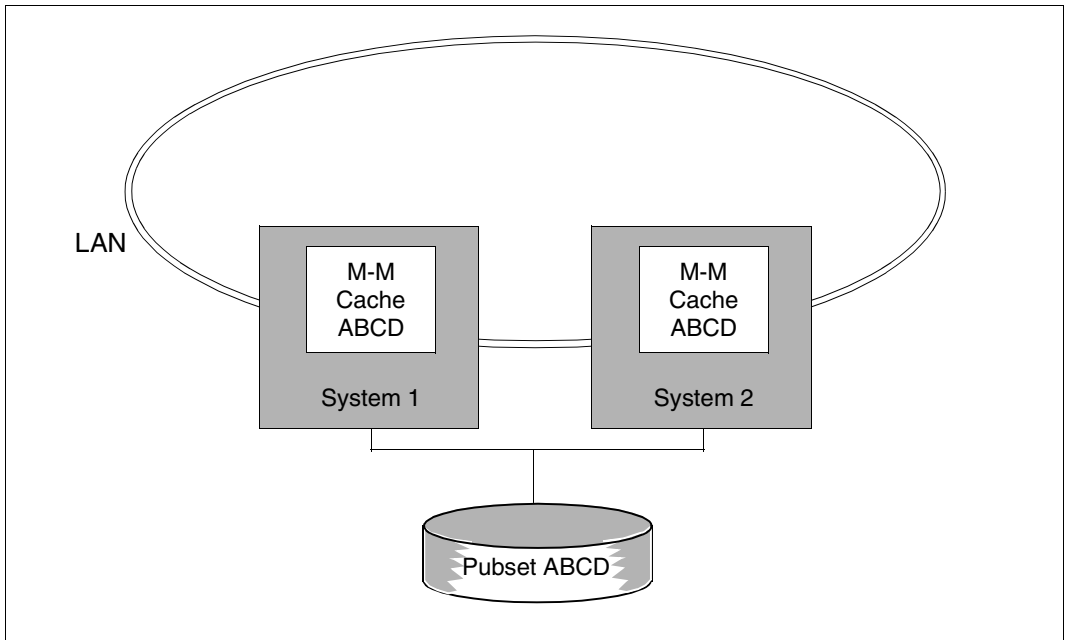


Figure 9: Shared pubset network with main memory

The use of the main memory medium is suited mainly to read caching, as it does not guarantee reliability in the event of a system crash. This mode is set by default for all permanent files by DMS. Write caching only takes place for a temporary file or if the user has explicitly assigned the attribute `DISK-WRITE=BY-CLOSE` to the file. The failure of a system is handled the same way here as with exclusively imported pubsets. The cache data of the failed system is lost. However, this does not matter since we are dealing with data from temporary files that have not been closed and where the data would be deleted automatically anyway.

If a shared pubset is to be buffered using DAB, the DAB subsystem must be loaded on all the participating computer nodes.

Caching in global storage

Global storage is (when using battery buffering, VOLATILITY=*NO) a non-volatile medium and is therefore suitable for use with write caching. DAB supports both local and global GS utilization, which must be clearly distinguished from one another:

Caching in the local GS

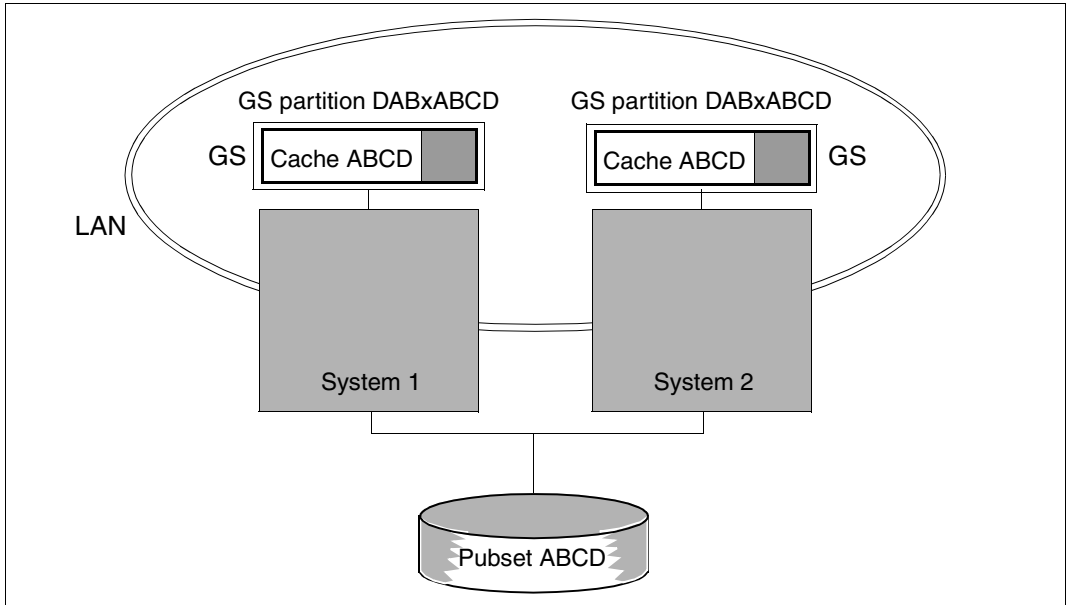


Figure 10: Shared pubset network with caching in the local GS

A GS is available on at least one pubset sharer for the utilization of a local GS configuration. A GS may also be available on multiple or on all pubset sharers. These global storage media are used locally, i.e. each server has its own GS, the servers do not form a parallel HIPLEX.

Systems support must set up a GS partition for a cache area on each server. The cache area is created on the server during import of the shared pubset, or it can be created subsequently with the `/START-PUBSET-CACHING` command. The local system cache area is released during export of the pubset or subsequently using the `/STOP-PUBSET-CACHING` command.

Please note the following:

Utilization of a local GS configuration is subject to restrictions with regard to reconfiguration in the event of a server crash or in the event of the connection between server and cached disks failing.

Since the GS can only be accessed from a server in a local GS configuration, the corresponding cache area cannot be transferred to the pubset master if this server crashes. Any files that were not closed properly at the time of the server crash and which used write caching (open mode=OUTPUT/INOUT/OUTIN) remain locked, as they have cache data on the crashed server. An attempt to open these files from a different pubset sharer is rejected with the DMS return code 0D1F, the status is displayed with the `/SHOW-FILE-LOCKS` command. These file locks are only reset when the crashed server is restarted and the pubset is reimported. During reimport, the GS cache area is reconnected to the pubset, the cache data is written to the disks and the file locks are reset. The files can then be accessed from all pubset sharers again.

Caching in the global GS (parallel HIPLEX)

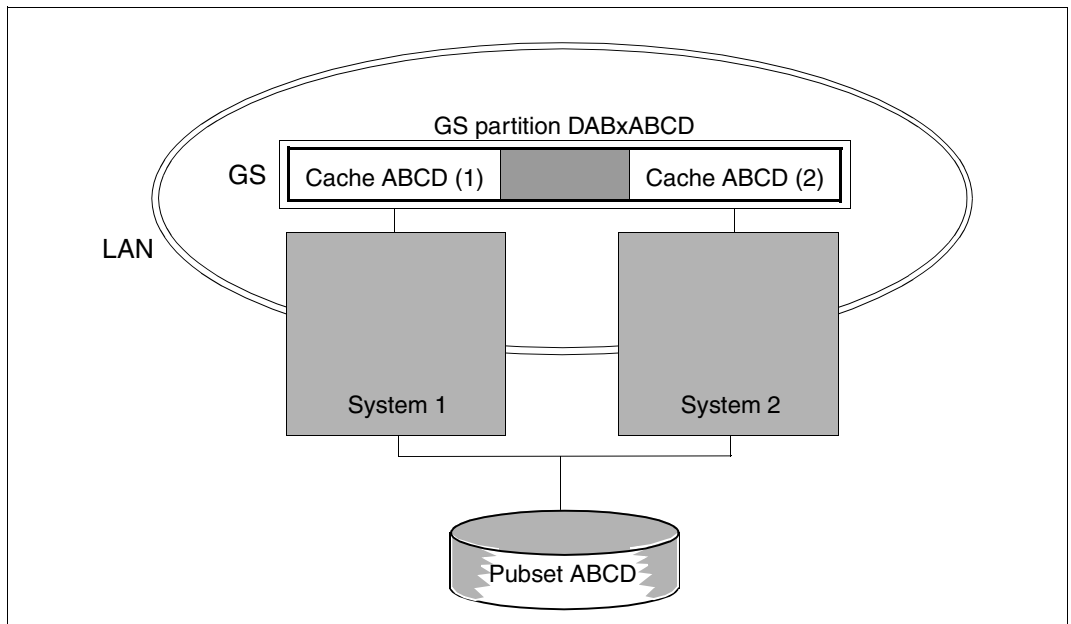


Figure 11: XCS network with caching in the global GS (parallel HIPLEX)

In a global GS configuration, all pubset sharers are attached to the same GS and form a parallel HIPLEX. Functions are made available which allow the GS to be managed centrally and utilized as a shared medium.

A further homogeneity condition applies here since in a parallel HIPLEX all DAB instances on the individual servers share access to a GS partition:

The same DAB version must be installed on all servers that wish to create a cache area in GS for a shared pubset.

Systems support must create a GS partition for use as a cache medium, with a size based on the sum of the sizes of all of the pubset sharer cache areas to be included (+ 1 MB for the DAB administration data). Cache areas are created in this GS partition on the relevant pubset sharers. The GS partition is divided internally into segments for the pubset sharers, corresponding to the cache size. The GS partition is thus shared so that all pubset sharers have access to the same GS partition.

If a server crashes or the connection between server and cached disks fails, a cache area of a pubset sharer can be reassigned to the pubset master in the case of a global GS configuration. In the event of a server crash (slave or master), this happens automatically within the context of failure detection by MSCF. The data of the reassigned cache area is written to the disks on the pubset master. The file locks, which until this time have prevented the cached files from being opened by pubset slaves, are then removed (on the pubset master the files can be opened after reassignment of the cache area, i.e. after reconfiguration of the shared pubset network by MSCF). The files can then be accessed from all pubset sharers again. For more information, see [section “Reconfiguration in the computer network \(parallel HIPLEX\)” on page 53](#).



The functionality described above for the failure of a server in the shared pubset network is only available in a parallel HIPLEX, utilization of the GS as a shared medium is recommended.

4.3 Creating and releasing cache areas

With ADM-PFA caching, a cache area is created via the `/START-DAB-CACHING` command, with PFA caching via the DMS interface (`/IMPORT-PUBSET` or `/START-PUBSET-CACHING` command).

With ADM-PFA caching, a cache area is released using the command `/STOP-DAB-CACHING` (or, in exceptional cases, `/FORCE-STOP-DAB-CACHING`). With PFA caching, the DMS interface is used to release a cache area (`/EXPORT-PUBSET` or `/STOP-PUBSET-CACHING` command) or, in exceptional cases, the `/FORCE-DESTROY-CACHE` command.

The option of simultaneous buffering of disk data via PFA and ADM-PFA is not available. Otherwise, `/START-DAB-CACHING` commands for data areas located on or belonging to pubsets buffered via PFA will be rejected. Similarly, any attempt to create a (USER-)PFA cache area is rejected if all or some of the associated disks of the pubset are buffered using ADM-PFA.

4.3.1 ADM-PFA caching

Creating

A new DAB cache area is created with the `/START-DAB-CACHING` command.

Releasing

The `/STOP-DAB-CACHING` command or, in exceptional cases, `/FORCE-STOP-DAB-CACHING` causes a DAB cache area to be released, with the following main action being performed:

With write or read/write caching, any data of the cache area to be released that is still in the cache is saved if necessary (unless the `/FORCE-STOP-DAB-CACHING` command has been issued). As a result, it may take some time to release the cache area using `/STOP-DAB-CACHING`.

4.3.2 PFA caching

PFA cache areas are created and released via the DMS interfaces and cannot be controlled via the DAB commands described in this manual. Refer to the “Commands” manual [4] for a description of the commands used for controlling PFA cache areas.

Creating

A PFA cache area is created via the DMS interfaces. Systems support defines a cache for a specific pubset (/MODIFY-PUBSET-CACHE-ATTRIBUTES command). The cache can then be activated in two different ways:

- implicitly during pubset import (/IMPORT-PUBSET command)
- during pubset operation (/START-PUBSET-CACHING command).



The GS partition for the pubset must be big enough to accommodate all required local system cache areas.

Releasing

There are various ways to release a PFA cache area:

- implicitly during pubset export (/EXPORT-PUBSET command)
- during pubset operation (/STOP-PUBSET-CACHING command)
- in exceptional cases (defective disk or memory) by means of the /FORCE-DESTROY-CACHE command (forced release).

Data held in the cache is saved to disk if necessary. Releasing with a /STOP-PUBSET-CACHING command may therefore take some time. However, the data is not saved if a cache area is released with a /FORCE-DESTROY-CACHE command and this can therefore result in data inconsistencies on the disks concerned.

4.3.3 Releasing cache areas during shutdown

When terminating the operating system (shutdown) DAB releases all existing cache areas, unless this process is prevented by an appropriate setting in the DAB subsystem parameters.

Special features for the cache medium GS

ADM-PFA cache areas with the cache medium GS are not released at system shutdown if the subsystem parameter KEEP-GS-CACHE-BUFFER-AFTER-SHUTDOWN = is set to YES. However, at least the data from the write cache areas are written back to the associated disk storage locations.

In combination with the subsystem parameter GS-READ-CACHE-BUFFER-RECOVERY = ENABLED the read cache areas in GS are restored the next time the system is started, this feature can be used to ensure that applications start up quicker.

The default setting for the subsystem parameter is KEEP-GS-CACHE-BUFFER-AFTER-SHUTDOWN = NO which triggers the release of all cache areas. Write data in cache areas in GS that cannot be written to disk during shutdown due to malfunctions will be retained. The cache areas will be restored during recovery of the corresponding DAB GS partition from the GS (see [page 51](#)). However, release of the cache areas by the systems support staff is advisable to enable them to take any action that may be required in the event of an error (e.g. disk failure).

4.4 Dynamic reconfiguration

4.4.1 Changing the configuration parameters of the cache area

The configuration parameters of a cache area (ADM and user PFA) can be modified dynamically using the `/MODIFY-DAB-CACHING` command. This means that it is no longer necessary to release the cache area explicitly using the `/STOP-DAB-CACHING` or `/STOP-PUBSET-CACHING` command and then recreate it using the `/START-DAB-CACHING` (specifying new parameters) or `/START-PUBSET-CACHING` command (after modification of the pubset cache attributes).

The following configuration parameters can be modified dynamically:

- **Cache size (CACHE-SIZE):**
The size of a cache area can be expanded or reduced. The cache is reduced without affecting cache utilization for the segment of the cache area that remains.
- **Cache segment size (CACHE-SEGMENT-SIZE):**
It is only useful to modify this configuration parameter for cache areas where there is no automatic caching (`AREA=*FILE` for ADM-PFA or `CACHED-FILES=*ALL/*BY-USER` for user PFA). All of the value specifications permitted with the `/START-DAB-CACHING` command are then possible. The cache size can only be modified quasi-dynamically because of the reorganization of the cache that is required as a result, i.e. caching of the data area is stopped, the cache is saved if necessary and an empty cache is added after reorganization of the cache. The configuration of the database to be cached is retained completely.
- **Adapting the cached data area (AREA=*ADD-FILE/*REMOVE-FILE/*RESET-FILE/*ADD-PUBSET/*REMOVE-PUBSET):**
The data area can only be changed for ADM-PFA cache areas.

`AREA=*ADD-PUBSET` adds a pubset which has not yet been buffered to a cache area using automatic caching. `AREA=*REMOVE-PUBSET` removes a pubset which has so far been buffered from a cache area using automatic caching.

`AREA=*ADD-FILE` adds files to a cache area and `AREA=*REMOVE-FILE` removes files from a cache area. A distinction must be made here between cache areas with and without automatic caching:

Cache areas without automatic caching:

Previously unserved files can be added to the data configuration, or previously served files which had a poor hit rate, for example, can be eliminated from the data configuration.

Cache areas with automatic caching:

The caching of files can be influenced directly without the monitoring algorithms of automated caching, i.e. files which are added to the data configuration are always cached or files which are removed are no longer cached.

AREA=*RESET-FILE resets modifications for files, and the files are once more cached automatically.

- Modifying the threshold for buffering (FORCE-OUT):
The threshold for the start of buffering for the cache can be set to one of three possible values: *NO, *AT-HIGH-FILLING or *AT-LOW-FILLING.
- Modifying the caching mode (CACHING-MODE):
The caching mode can only be modified for ADM-PFA cache areas.
There are two different “directions”:
 - When changing from *READ to *READ-WRITE/*WRITE/*BY-CACHE-MEDIUM, the data imported in read mode is retained and is processed further in the new mode concerned from this point on.
 - In the other direction (to *READ), data is written to the disk(s) without a disk copy; only read data is then imported in the cache.

4.4.2 Modifying the disk allocation or the entries in the file catalog

Initial allocation or final release of a disk

During initial allocation of the disk concerned, a check is performed as to whether allocation and set cache attributes permit caching

(command `/START-DAB-CACHING . . . , SHARED-DISK-SUPPORT=*YES/*NO`; see [page 38](#)).

When a disk is released (deallocated), all the released data is saved and invalidated in the cache. Invalidation ensures that no data of the disk concerned is read from the cache once the disk has been detached and reattached.

Allocation or release of disk space for a file

The data area served by DAB is automatically extended or reduced whenever disk space is allocated or released for a file served by DAB. This ensures that a file will be served by DAB to the full extent following storage space allocation or to the available extent following storage space reduction.

Furthermore, the associated cache area is extended if the data areas of an ADM cache area are to be served using “resident buffering”

(command `/START-DAB-CACHING . . . , CACHE-SIZE=*BY-FILE`).

If, when processing data accesses or when adapting the DAB configuration to the modified storage requirements of a file, the system discovers that the DAB configuration is no longer up-to-date as a result of a previous system error and cannot be corrected using the information currently available, then the data located in the cache for the affected file is written back and this file is removed from the current cache configuration (see message NDB0145). The table shown below can be used to help resume caching of a file:

Cache class	Measures
ADM-PFA cache areas with AutoDAB	None The affected file is entered back into the cache configuration at the next data access <code>MODIFY-DAB-CACHING AREA=*RESET-FILE(...)</code>
ADM-PFA cache areas with <code>AREA=*FILE (...)</code>	<code>MODIFY-DAB-CACHING AREA=*ADD-FILE(...)</code>
User PFA cache areas	<code>START-FILE-CACHING</code>

Table 5: Measures for resuming file caching

Recataloging/deleting files served by DAB

DAB detects when a buffered file is renamed and updates the associated management data accordingly. Renamed files continue to be buffered by DAB.

If a temporary file is converted to a permanent one for ADM-PFA cache areas with automatic caching (AutoDAB) in the main memory cache medium and CACHING-MODE=*BY-CACHE-MEDIUM, all write data of this file is saved if necessary. The file is subsequently only served with read caching.

The management data associated with any file that is deleted during DAB operation will be deleted in the corresponding cache area.

If a new file is created and assigned the name of the deleted file, it can be buffered in the previous cache area by using the command

`/MODIFY-DAB-CACHING . . . ,AREA=*ADD-FILE(<filename>)`. Alternatively, the file can be buffered in a new cache areas with an additional `/START-DAB-CACHING` command with `AREA=*FILE`.

In the case of a cache area with automatic caching, the next time it is opened it is reentered into the configuration of the existing cache area of the pubset or private disk.

When files buffered in an ADM-PFA cache area with `AREA=*FILE` and `CACHE-SIZE=*BY-FILE` (resident buffering) are deleted, the cache area may be larger than required to serve the remaining files. It is up to systems support to check whether the space occupied by the cache is used economically and to reduce the cache area if required (see the `/MODIFY-DAB-CACHING` command on [page 89](#)).

4.5 Restoring DAB cache areas in GS

The cache medium global storage enables DAB to continue using cache areas after an interrupt without losing any data that was buffered in the cache but not yet saved on an external volume at the time the interrupt occurred.

Such an interrupt may be caused, for instance, by the following:

- abnormal termination of the operating system with subsequent system restart on the same or another server in the case of local GS utilization
- reconfiguration in the parallel HIPLEX as a result of abnormal shutdown of a server in the network or the failure of the physical data paths between a server and the external data media required by the application in question

Read caches are restored only if `GS-READ-CACHE-BUFFER-RECOVERY=ENABLED` is set in the DAB subsystem parameters.

4.5.1 Steps involved in cache area restoration

Cache areas are generally restored during disk allocation. This ensures that the data is available before processing with the files can continue.

In the case of public volumes, disk allocation takes place during the pubset import, for private disks it occurs during referencing of a file on the disk (or explicitly with the command `/SET-DISK-PARAMETER <mn>,ASSIGN-TIME=*OPERATOR`).

Restoration of a cache area is performed in the context of restoring all data in a DAB-GS partition, i.e. it causes all cache areas in the same GS partition to be restored and the GS partition to be cleaned up.

ADM-PFA cache areas that are configured as read cache areas are restored in so far as this is set in the DAB subsystem parameter `GS-READ-CACHE-BUFFER-RECOVERY=ENABLED`. This setting must be entered in the associated GS partition at both the time that the affected cache is set up as well as the time of the restoration. The default of the subsystem parameter at delivery is `GS-READ-CACHE-BUFFER-RECOVERY=DISABLED`, thus read cache areas are not restored.

The restoration of ADM-PFA cache areas serving data areas of the current home pubset constitutes a special case: since the volumes of the home pubset are already allocated at system startup, i.e. before the subsystems DAB and GSMAN are available, the corresponding cache areas cannot be restored at the time of volume allocation.

In the case of local GS utilization, the restoration of cache areas for the home pubset occurs automatically when the DAB subsystem is started during system startup.

See [section “Restoring cache areas on servers with local GS utilization” on page 52](#) for a description of this special case and the way it is handled.



Despite this automatic restoration of cache areas for data of the home pubset, it is not permitted to serve the data areas of the home pubset, which are accessed before “System Ready”, in write or read/write caching mode.

In a parallel HIPLEX, the GS-specific functions of the GSMAN subsystem are only available after “System Ready”, which means that the restoration of cache areas is only possible after this time. It is therefore vital to restore cache areas that serve the data of the home pubset through write caching as soon as possible. Caching of the home pubset through write caching is therefore not supported in a parallel HIPLEX, a corresponding /START-DAB-CACHING command is rejected.

4.5.2 Restoring cache areas on servers with local GS utilization

In such a system environment, cache areas cannot be restored until the operating system has been restarted on the same server. Cache areas for data not located on the current home pubset are restored at initial allocation of the volumes, thus ensuring the availability of the cache data before the first access.

For home pubset data that was served by ADM-PFA caching in write or in read/write caching mode in the preceding caching session, the cache areas are restored automatically when the DAB subsystem is started during system startup. This action is completed prior to “System Ready”.

Files which are accessed before “System Ready” must not be served with write or read/write caching.

4.5.3 Reconfiguration in the computer network (parallel HIPLEX)

With DAB it is possible to continue using a cache area from another system of the parallel HIPLEX after an interrupt.

This is useful if the cache contains as yet unsaved data and the cached disks are to be reassigned to another system after the interrupt (or are already shared there).

A reconfiguration in the computer network may be required in the following cases:

- server crash as a result of a hardware or software error
- failure of the connection between the server and the disk.

The reconfiguration options as well as the points to be observed during reconfiguration are explained in the server configuration illustrated below. In this configuration an exclusively used DAB-GS partition (partition A and B) is created for each server; the data areas of the connected disks are cached in this partition. Both servers use a shared DAB-GS partition (partition C), in which data areas of a shared pubset are cached.

The possible situations are explained in detail below.

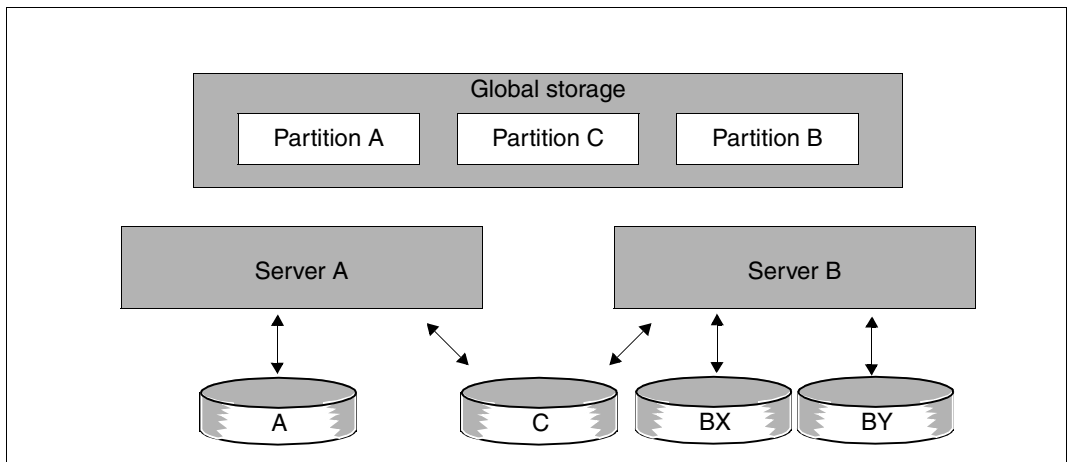


Figure 12: Server configuration example

4.5.3.1 Cache areas for exclusively allocated disks

These may be PFA cache areas for exclusively imported pubsets or ADM-PFA cache areas for exclusively allocated disks (pubset or private disk).

Server crash as a result of a hardware or software error

If there is a server crash (server B) where there are cache areas with write caching for exclusively occupied disks, the disks can be reassigned to another server in the parallel HIPLEX (server A).

For the first allocation of a disk (e.g. with the `/IMPORT-PUBSET` command), the corresponding GS partition is automatically reassigned to this server. Reattachment of the GS partition on server A is allowed after receipt of message EGC0503 confirming that server B can no longer access the GS partition.

In the context of this reassignment, all cache areas contained in the GS partition are restored by DAB, with the result that the data is available consistently again from the cache when these disks are accessed. The reattachment unit in this case is the GS partition, i.e. all disks that are buffered with cache areas in a GS partition must be reassigned to the same server in the parallel HIPLEX.

Failure of the connection between the server and the cached disks

If the data paths to the cached disks fail, you have the option of reassigning the disks and the corresponding GS partition of the cache areas to another active server in the parallel HIPLEX. The reattachment can take place when it has been ensured that the GS partition to be reassigned can no longer be accessed from the first server.

In a normal situation, it is always possible to end utilization of a GS partition by releasing the cache areas. However, if the data paths to the cached disks fail, a normal release is not usually possible because the cache data cannot be saved. There is therefore an option to disconnect the cache areas on the first server from the GS partition.

DAB performs a disconnection for ADM-PFA cache areas if writing of the cache data with a `/STOP-DAB-CACHING` command is unsuccessful and the appropriate response is given to the subsequent NDB0031 message. The cached disks must no longer be occupied at the time of the `/STOP-DAB-CACHING` command, i.e. pubsets must have been exported and occupation of private disks must have been stopped with the command `/CHANGE-DISK-MOUNT UNIT=<mn>, ACTION=*CANCEL`.

With PFA cache areas, disconnection takes place automatically when the corresponding pubset is exported if the cache data cannot be written correctly.

A GS partition is (completely) disconnected from a server when all previously existing cache areas have been either released or disconnected. A disconnected GS partition can then be reassigned to another server in the same HIPLEX, i.e. the cache areas still remaining in the GS partition can be restored from another server.

In order to completely disconnect a GS partition from a server (server B), any cache areas still available in this partition must be released.

The reattachment of the GS partition takes place, as is the case when a server crashes, during the first allocation of a cached disk to another server in the parallel HIPLEX. The GS partition is started on this server and all of the cache areas contained in this partition are restored.

Summary of required measures

User PFA:

- /EXPORT-PUBSET on system B
- /IMPORT-PUBSET on system A

ADM-PFA:

- /EXPORT-PUBSET on system B for data on pubsets or
/CHANGE-DISK-MOUNT <mn>, ACTION=*CANCEL for data on private disks
- /STOP-DAB-CACHING for the cache areas of the GS partition on system B
- Respond to the message NDB0031 on system B with 'Y'
- /IMPORT-PUBSET on system A for pubsets or
/SET-DISK-PARAMETER <mn>, ASSIGN-TIME=*OPERATOR for private disks



- The sequence of the /STOP-DAB-CACHING commands does not matter here; a GS partition is always disconnected by DAB if it contains a disconnected cache area and the last cache area of this partition was released or disconnected with the current /STOP-DAB-CACHING command.
- The disconnected cache areas can be restarted on the same server, e.g. after the failed data paths have been restored.

4.5.3.2 PFA cache areas for shared imported pubsets

GS partitions for PFA cache areas which serve shared imported pubsets are divided into segments, or slices, for the servers using them. In the event of a server crash (or reassignment because of the loss of a connection to the disks), only the slice of the GS partition that is assigned to the server is reattached and the cache area contained in this slice is then restored. The reassignment always takes place on the pubset master or, if the master crashes, on the newly established master (master reattachment).

Server crash

If a pubset slave crashes (e.g. server B), the cache area used by the slave is reassigned to the pubset master, reassignment taking place automatically within the context of slave failure detection by MSCF. If the pubset master crashes, its cache area slice is reassigned automatically as part of the master reattachment performed by MSCF. In both cases, the cache area of the GS partition slice assigned to the server is restored on the pubset master and the cache data is written to the disks. Once the cache data has been saved, the file locks (which prevent the relevant files from being opened by any server other than the master) that have been in place until now are reset. The files can then be accessed from all servers in the shared pubset network again.

Failure of the connection between the server and the cached disks

If the physical connection to the disks fails for a shared pubset while the network connection via BCAM is still intact, MSCF does not detect a crash. The status is detected by the monitor, but the shared pubset is not exported (unlike in previous versions, where a compulsory export was initiated by the system).

If this error occurs on a pubset slave, files can still be opened (metadata, such as that in TSOSCAT for example, is accessed on the master). However, it is no longer possible to read data or write cache data to disk. In some situations, the appearance of DMS return code 0E27 when a file is closed indicates that an I/O operation could not be performed in the course of closure.

If the pubset on the slave is exported by systems support, the GS partition slice assigned to the server is disconnected during the export. The slice is then reassigned to the pubset master and the cache area of the GS partition slice is restored there. As is the case with reassignment after a server crash, this is followed by the writing of the cache data and the resetting of any file locks that may still be in use.

If a connection to the disks on the pubset master fails, metadata can no longer be accessed, i.e. no files on the shared pubset can be opened or closed (even on the pubset slaves). The cache data of the files opened on the master cannot be written to the disks either. In order to rectify this situation, an `/EXPORT-PUBSET` command with master reattachment (operand `SHARER-TYPE=*MASTER(MASTER-CHANGE=*YES)`) must be initiated on the pubset master.

In the context of the master reattachment, the GS partition slice used by the previous master is automatically disconnected and reassigned to the new master, where the cache area it contains is then restored. As is the case with reassignment after a server crash, this is followed by the writing of the cache data and the resetting of any file locks that may still be in use. The cached files can then be accessed from all pubset sharers again.

Notes on reconfiguration in the computer network (parallel HIPLEX)

When planning the GS cache configuration in a HIPLEX-MSCF network, the following rules must be observed in order to enable reconfiguration in the network:

- The GS partition forms a “reattachment unit”; all data areas buffered here must be reassigned to the network nodes as well (see [section “Structure of global storage” on page 24](#)).
- In the case of a reconfiguration, cache areas are restored on a node in a HIPLEX-MSCF network. It must be ensured that the required resources (especially the resident main memory for the administration tables of the cache areas) are available. It may be necessary to first release existing cache areas in order to free up the required memory. As a result of the reconfiguration options available in a parallel HIPLEX, a system configuration, which allows shared operation of all cache areas in the XCS on each node, must be made available on each server.
- It must be ensured that the real main memory needed to operate the cache areas is available.

5 Notes on DAB usage and performance behavior

This chapter contains the following:

- information about how to use DAB efficiently with the read, write and read/write caching modes
- basic notes concerning DAB usage
- information on using DAB in conjunction with other products
- the “Performance behavior” section describes the ways in which file access times can be improved, how to relive the pressure on the I/O system and how your choice of cache media affects the CPU requirements.

5.1 Efficient use of DAB

The performance gain attained through caching I/O accesses is subject to different conditions, depending on the caching mode used.

5.1.1 Read caching

The performance gain attained through read caching is dependent on the achieved (read) hit rate. Data requests which can be served directly from the cache are handled very fast when compared with disk accesses.

The hit rate is in turn dependent on access locality, the size of the cache area available and the prefetching factor. In the last analysis, the interaction of these three parameters is decisive for read cache efficiency. This is illustrated by the examples given below:

- Sequential file processing
In many applications, files are processed by a series of sequential data accesses, i.e. the processing sequence matches the sequence of the data on the disk. Favorable hit rates are obtained automatically for data accesses by prefetching major sets of data and holding them in the cache.
- Frequent read accesses to selected data areas
In many applications, access to special data areas is particularly frequent (e.g. catalogs, index areas and directories). In these cases, buffering of this data in a fast storage is advisable to speed up access to the data (this principle is taken into consideration by DAB). This also considerably reduces the overall number of disk accesses, with positive effects on system performance.

5.1.2 Write caching

The file access times for write requests can be shortened considerably by using either the write caching or write/read caching mode.

DAB accelerates any write access processing, irrespective of data locality, provided write caching or read/write caching is enabled and free segments are available. Caching ensures that the data is written from cache to disk asynchronously from the I/O job. With good locality, this significantly contributes to reducing the I/O system load.

Write caching offers the following advantages:

- From the point of view of the requesting programs, the file access times can be shortened considerably since the data to be transferred is first moved to the cache. As far as the application is concerned, the write request has been completed. The transfer of the buffered data from cache to disk does not normally affect program runtime, and the user can specify during cache area definition (`/START-DAB-CACHING` or `/MODIFY-PUBSET-CACHE-ATTRIBUTES` command, `FORCE-OUT` operand) at which point the transfer occurs.
- The I/O system load is reduced:
In the event of good spatial or temporal locality of the buffered data, buffering and asynchronous writing of the data to disk results in the number of disk accesses actually performed being smaller than the total number of data transfers initiated by the programs.
With a suitably large cache, the transfer of buffered data from cache to disk before the application is closed can be omitted entirely, which minimizes the I/O system load. Subject to good temporal locality of the buffered data, read accesses following the transfer of the data to the cache can be satisfied from the cache efficiently.

Using the caching technique “resident buffering” effectively precludes cache overflows, provided the cache size matches the size of the area it serves, i.e. a sufficiently large cache area must be created. With ADM-PFA caching (operand `CACHE-SIZE=*BY-FILE`), the dynamic growth of the cache area must be noted and the resulting free cache space required must be made available.

5.1.3 Read/write caching

The comments on read caching and write caching by analogy also apply to read/write caching. With read/write caching, however, conflicts between write data and read data may arise within the available cache area: for instance, a heavy I/O system load may result in the transfer of write data to disk being delayed for some time. This causes more and more of the storage space available in the cache to be filled with write data, proportionally reducing the space available for read data and consequently the number of read hits. Good hit rates should therefore always be of special concern.

With the caching technique “resident buffering”, the situation just described can never arise, since the cache size matches the size of the area it serves.

5.1.4 Read caching of encrypted files

DAB supports the caching of encrypted files in order to optimize processing of this class of file. However, only read caching of encrypted files is supported, no write caching. Without caching, the access times when processing encrypted files are - because of the use of encryption methods - higher than when processing unencrypted files. When caching is used, DAB can further optimize the processing of encrypted files as follows:

- Reduction of the mean I/O access time for encrypted files by dispensing with encryption operations in the event of cache hits:
 - The I/O times can be optimized through caching.
 - No encryption operations need be performed for data transfers, which can be executed as cache hits. Consequently caching can also result in a considerable reduction of the I/O access times when processing encrypted files.
 - Encryption operations which noticeably extend the I/O path. If the accesses to an encrypted file demonstrate a good temporal locality, a large proportion of the data accesses can be performed without encryption operations.
- Reduction of the mean I/O access time for encrypted files by bundling several encryption operations:
 - When data is read from disk into the cache or when the cache is saved, encryption operations must always be performed. Frequent referencing of the same data in the cache rarely occurs.
 - Consequently dispensing with encryption operations by means of cache hits brings little overall benefit in terms of I/O access times to encrypted files with mainly spatial locality.
 - Separate execution of the encryption operations is made more efficient by bundling the encryption of multiple disk areas in one encryption operation.
 - The bundling of encryption operations goes hand in hand with the creation of long I/O chains, as is possible for caching with good spatial locality.
 - This more efficient bundling of data encryption means that caching with DAB nevertheless reduces the mean I/O time.

The same interfaces and functions are available to users for the read caching of encrypted files as is the case for unencrypted files.

5.2 Notes concerning DAB usage

The following points apply in general:

- The selection of the database to be buffered and the prefetching factor setting should be done via automatic caching. It is therefore advisable to serve performance-critical data storage media with AutoDAB. This is done ideally with a single `/START-DAB-CACHING` command with which the volumes concerned are specified and all available cache memory is used.
- The SM2 software product (see “openSM2” manual [7]) can be used to monitor DAB efficiency during operation and to initiate any tuning measures that may be required.

The following points are relevant when using DAB:

1. Write caching is not allowed for any files of the home pubset accessed by BS2000/OSD before 'System Ready' (e.g. TSOSCAT, JOIN or SYSSRPM files, REP files etc.).
2. When caching volumes with high availability (DRV, RAID1, RAIDS) it must be ensured that availability is not reduced by using the cache. Pure read caches are uncritical in all cases. Write caching of critical data is only advisable in GS caching media using dual recording (GS-UNIT=*DUAL).
3. DAB cannot be used to buffer the paging area.
4. For data security reasons, write caching of file catalogs (TSOSCAT) does not make use of the volatile cache media main memory, even if the associated cache area has been activated in write mode.
5. When a disk served by DAB is detached by means of the command `/DETACH-DEVICE . . . ,FORCE=*YES`, any data on this disk that may still be contained in the cache will not be saved. However, the buffered disk data continues to be held in the cache and will be available again with the next allocation of the disk on the same system.

Furthermore, the following differences exist depending on the cache medium used:

- Main memory (MM):
Reattaching the disk to another system is likely to result in data loss since cache areas of this type cannot be reassigned to another system. In addition, if the disk is subsequently reattached to the original system again, any cache areas on this computer that serve data areas of the disk must first be released by means of the `FORCE-STOP-DAB-CACHING` command to prevent any data on this disk that may still be held in the cache from being used.

- Global storage (GS):
With global storage, the disk may be reattached to another system together with the associated cache area to continue processing of its data on the new system. Prior to reattachment, all cache areas of the GS partitions concerned must be either disconnected or released. There is no risk of data loss in the event of the disk being subsequently reattached to the original system.
6. All assigned cache areas must be terminated before formatting disks (VOLIN utility) or regenerating pubsets (SIR-IP utility).
 7. When buffering a file that has been assigned the (cache) performance attribute “VERY-HIGH”, all cache segments allocated to this file are locked against displacement until the file is closed. If a large number of files with this performance attribute are open simultaneously, the cache space available for buffering ordinary files may be reduced to the extent that access to the data of these files may become unacceptably slow.
 8. Since the cache areas that can be created in the main memory and global storage media can be very large, more time must be allowed for saving the cache data when releasing large cache areas used in read or read/write mode. This applies for both ADM-PFA and PFA caching.
 9. Refer to [section “DAB in multiprocessor operation” on page 37](#) for notes on caching with shared private disks or shared pubsets.
 10. GS cache areas created with another DAB version cannot be reconstructed. All DAB cache areas in GS and the DAB GS partitions must be released prior to a version change.
 11. Cache areas in main memory (MM) are created as resident storage when the operand `MEMORY=*STD/*BELOW-MIN-MEM-SIZE` is specified in the `START-DAB-CACHING` command. It must be ensured that the main memory is adequately dimensioned according to the cache size. In the case of operation on a virtual machine, its `MINIMAL-MEMORY-SIZE` must also be dimensioned appropriately.
 12. If, in a cache area using automatic caching, cache overflows occur over several monitoring periods or too many files have to be excluded from caching due to their poor cache utilization, systems support is informed via console message NDB0109 that the cache area is too small for the current load. The cache area should be increased in size with the `/MODIFY-DAB-CACHING` command in this case.

Switching between cache areas for ADM-PFA and PFA caching

DAB does not support the coexistence of ADM-PFA and PFA cache areas on the same disk. This means that no ADM-PFA cache area can be created for the data areas of a disk once it has been buffered via PFA, and no PFA cache area can be created for a pubset as long as at least one disk of the pubset is (even partially) buffered via ADM-PFA. However, changing the caching methods for a disk is possible, subject to the following constraints:

- ADM-PFA caching → PFA caching:
A PFA cache area can be defined for a pubset once all ADM-PFA cache areas serving any data areas of the pubset have been released.
- PFA caching → ADM-PFA caching:
An ADM-PFA cache area for data areas of a pubset cannot be created until no PFA cache area is active for this pubset. This means that an active PFA cache must be released (e.g. via a `/STOP-PUBSET-CACHING` command) prior to installing an ADM-PFA cache area. The PFA caching definition for this pubset must also be deleted explicitly. This requires the following actions:
 1. PFA caching must be stopped with the `/STOP-PUBSET-CACHING` command.
 2. Specification of the cache medium in the pubset-specific parameter records should be changed by setting the `/MODIFY-PUBSET-CACHE-ATTRIBUTES` command to the value `CACHE-MEDIUM=*NO-CACHE`.

5.3 Using DAB in conjunction with other products

If DAB is to be used in conjunction with one or more of the following products, you should ensure that you bear the following information in mind.

5.3.1 DAB and DRV

Where requirements for data security are particularly stringent, the use of DRV (dual recording by volume) is recommended. DRV implies that data is stored and maintained in a consistent state simultaneously on two physically separate disks (redundant data maintenance). The two disks (referred to as “mirror disks”) are operated as a single logical disk.

With PFA caching, write caching is only carried out for files if cache areas for DRV pubsets are located in nonvolatile, redundant cache media (GS with battery backup and GS-UNIT=*DUAL or CONTROLLER), i.e. “mirrored” files can be declared as HIPERFILES without adversely affecting the data availability.

As already mentioned in [chapter “Cache media” on page 21](#) is the only DAB cache medium that can be upgraded to a nonvolatile (failsafe) cache medium. Only the nonvolatile GS variant in dual mode (with write data being simultaneously buffered in two independent GS units) should be used for buffering DRV data (command /START-DAB-CACHING . . . ,CACHE-MEDIUM=*GS(GS-UNIT=*DUAL)).

5.3.2 DAB and GSVOL

GSVOL is a subsystem that provides disk emulation in global storage (similar to the principle of resident buffering).

Data stored on emulated disks cannot be buffered via DAB.

5.3.3 DAB and concurrent copy

The utilization of local write caches on a shared pubset is also permitted if a file is saved with concurrent copy.

The following restriction applies:

If the cache data of a file cannot be correctly saved to disk when the file is closed and the cache data of the shared pubset is on a slave system, the pubset master rejects the start of a concurrent copy session for this file.

5.3.4 DAB and external disk storage systems

The disk storage systems served by BS2000/OSD (ETERNUS DX, Symmetrix, CLARiiON CX) are high availability disk storage systems which fulfill both the old RAID standards of Berkeley University and all availability levels defined by the RAID Advisory Board.

These disk storage systems are protected against catastrophic events by local or remote replication functions, see the “SHC-OSD” manual [6].

Using RAID1, RAIDS or SRDF

When volumes in external storage systems are operated using DAB, it must be ensured that the increased data availability is also retained with caching (similarly to using DRV).

(Read/)write caching should only be employed in dual mode in two GS units with power failure protection. Read caching is never critical in this respect since the data is both in the cache and on the data storage media.

Local mirroring in storage systems

Local mirroring in storage systems provides the option of creating (additional) copies of the units within a disk storage system which can then be split and processed separately. This allows the original data to be made available to the main application while simultaneous backup and batch evaluations (which normally require that the application be terminated or interrupted) are carried out on the copy.

See the manual “SHC-OSD” [6] for more information.

When separating this local mirror pair, it must be ensured that the data of a (read/)write cache has been saved to the disk with DAB. This is ensured if the following conditions are met:

- The pubset was correctly exported, thereby producing both file and cache status consistency.
- The DAB cache areas were released before separation in the imported state with the `/STOP-DAB-CACHING` or `/STOP-PUBSET-CACHING` command.
- When separating in the imported state, you are able to change the existing DAB cache areas from (temporary) write mode to read mode using the `/MODIFY-DAB-CACHING` command. If only read cache areas are available for the disks, SHC-OSD allows you to separate a local mirror pair. After separation, the cache areas can be reactivated for write caching using the `/MODIFY-DAB-CACHING` command.



Separating local mirror pairs in the imported state using the option described above avoids the time-consuming build up of cache data that is required when releasing and recreating the cache areas. This means that you can continue working with optimal cache performance immediately.

5.3.5 DAB and SPACEOPT

DAB and SPACEOPT are mutually compatible. This means that it is possible to carry out storage optimization while the data volume that is being optimized is cached with DAB. It is not necessary to release cache areas for the data volume that is to be optimized before SPACEOPT is started.

Opened files which are buffered with user PFA are still not included in the reorganization by SPACEOPT (even if the inclusion of opened files is explicitly requested).

If an optimization run is interrupted as a result of system crash, then, when restoring the affected GS cache areas, DAB ensures that the relevant cache configuration is adapted to match the current storage space occupancy of the cached files.

5.4 Performance behavior

The following measurement results highlight the degree to which I/O behavior can be improved with the aid of DAB.

5.4.1 Improving I/O time

The I/O time is the mean operating time for an I/O job. With openSM2, the I/O time can be output on a file-specific basis via the measurement program FILE-STATISTICS (see the “openSM2” manual [7]). The I/O time generally consists of the following measurements:

$$\text{I/O time} = \frac{(\text{RD}_{\text{hit}} + \text{WR}_{\text{hit}}) * t_{\text{hit}} + \text{RD}_{\text{miss}} * t_{\text{GB}} + (\text{RD} + \text{WR}) * t_{\text{disk}}}{\text{total number of I/O jobs}}$$

where:

RD_{hit} Number of read hits.

WR_{hit} Number of write hits (only for write or read/write cache).

t_{Hit} Duration of a hit (0.1 ms to 0.2 ms depending on CPU and cache medium).

RD_{miss} Number of read misses (only for read or read/write cache).

t_{GB} Duration of segment caching, generally dependent on disk type and load as well as segment size.

A basic requirement (irrespective of the segment size) of 10 ms should be assumed for direct disk drive access, plus 0.2 to 0.3 ms per 2 KB transferred (see also the “Performance Handbook” [5]).

RD Number of read accesses to disk (only for write cache).

WR Number of write accesses to disk (only for read cache).

t_{disk} Mean disk access time = mean positioning time and rotational delay + transfer time (the transfer time being largely dependent on the number of blocks transferred).

To improve I/O time, a minimum read hit rate, dependent on the segment size, should be achieved (see also [figure 13 on page 71](#)). File access times will always be improved for write jobs, provided write caching is enabled and free DAB segments are available.

Detailed evaluation

A) Read cache

The I/O time for read jobs is essentially determined by the hit rate. To improve the I/O time, a minimum hit rate, dependent on the segment size, should be achieved (see the figure below).

where:

RD_{hit} is dependent on the cache size set, the access behavior and the segment size selected.

$WR_{hit} = 0$ Write jobs are always executed on disk (WR). If the block concerned is in the cache, it is updated there as well.

RD_{miss} A miss causes segment caching to be initiated.

$RD = 0$ Read jobs are always served from the cache area.

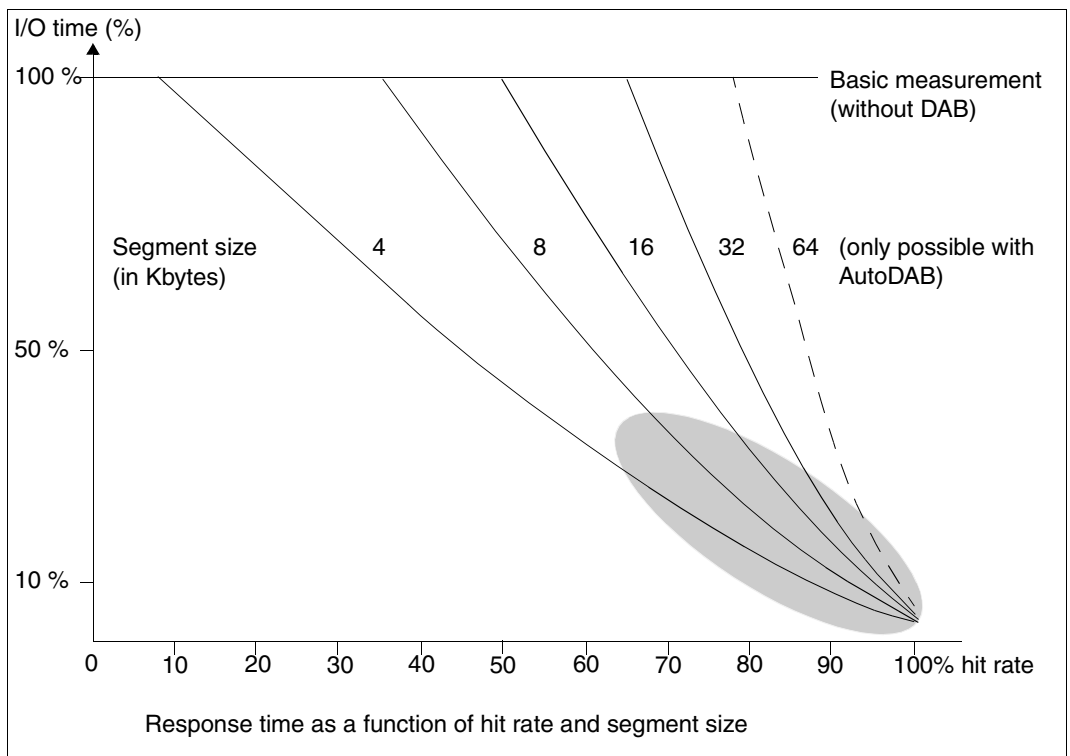


Figure 13: I/O times using the read cache (100% read share, random access, 2KB block size)

The [figure 13 on page 71](#) shows how the I/O time is reduced by DAB. The reduction is relative to the I/O time of the data media with direct disk drive access (e.g. approximately 11 ms for 3421, approximately 0.5 to 3.5 ms for a Symmetrix disk storage system with 100% or 70% overall hit rate). The hit rate and therefore the I/O times with AutoDAB are generally within the shaded area.

B) Write cache

With write jobs, the segments to be written are always transferred to the DAB cache area first, thus ensuring an excellent I/O time. Depending on the FORCE-OUT operand (see [page 73](#)), the DAB task asynchronously saves written segments to disk, attempting to chain several segments. With a very small cache area, there may be a temporary lack of free segments; in this case, the segments are written directly to disk (cache overflow).

where:

RD_{hit} is dependent on the cache size set, the access behavior of write jobs and the segment size selected.

WR_{hit} Segments to be written are always transferred to the cache first, provided there are free cache blocks available. If the cache is large enough, the write hit rate will therefore always be 100% ($WR = 0$). The DAB task asynchronously saves written segments to disk, attempting to chain several segments.

$RD_{miss} = 0$

A read miss causes the data to be read from disk without being cached, i.e. a normal read access (RD) to disk is executed.

C) Read/write cache

Again, all write jobs are accelerated here. This results in improved I/O time even with relatively low read hit rates. Again, I/O time improvement is dependent on hit rate and segment size.

where:

RD_{hit} / WR_{hit} are dependent on the cache size set, the access behavior and the segment size selected.

$RD = 0$ see A) read cache.

$WR = 0$ see B) write cache.

The behavior shown in [figure 13 on page 71](#) also applies to the read/write cache. The values for the read/write cache are close to those for the pure read cache in the highlighted area.

5.4.2 Reducing the I/O system load

The DAB hit rate and the segment size selected or the current prefetching factor are decisive for the extent to which the I/O system workload is reduced. Note that 2 / 4 / 8 / 16 / 32 PAM pages are transferred at a time in the event of segment caching caused by a read miss. This will reduce channel utilization with a good hit rate but increase channel utilization with a poor hit rate.

AutoDAB generally creates a cache behavior with low cache miss rates, resulting in a noticeable reduction of channel loading.

5.4.3 CPU utilization with the various cache media

The choice of the cache medium has a marked effect on CPU requirements and, to a lesser degree, on I/O times. Additional CPU requirements are lowest with cache areas in main memory (MM).

Where files with key access are buffered in global storage (GS), the key always has to be transferred to the cache medium, which means that an additional access to the cache medium is necessary. For non-key disks, this additional GS access is not required, thereby reducing I/O time and CPU requirements.

With various servers, using GS and a 100% hit rate shows no difference in CPU requirements as compared to the use of main memory.

Hit rates > 90% are generally achieved with AutoDAB. A small prefetching factor is set for accesses which are predominantly non-sequential (random) while a larger one is set for predominantly sequential accesses. The CPU utilization for the read/write cache includes the requirement for save I/Os to the disk.

Effect of the FORCE-OUT operand

The FORCE-OUT operand of the `/START-DAB-CACHING` command can be used to control saving of cache data to disk. `FORCE-OUT=*AT-LOW-FILLING` causes saving to be started whenever 25% of the cache area are filled with write data not yet saved; `FORCE-OUT=*AT-HIGH-FILLING` triggers saving at an occupancy of 75%. `FORCE-OUT=*AT-LOW-FILLING` is always recommended. `FORCE-OUT=*NO` is recommended for resident buffering, since this specification suppresses I/Os for saving data to disk, which is superfluous with this caching technique.

With automatic caching, any incorrect setting of this operand is corrected (see also [section "Automatic caching \(AutoDAB\)" on page 34](#)).

5.4.4 SM2 monitoring reports for DAB caching

The recording of measured values for ADM-PFA or PFA caching is started under SM2 by means of `//START-MEASUREMENT-PROGRAM TYPE=*DAB` or `TYPE=*PFA`, respectively. The recording of file-specific access times is started by means of `//START-MEASUREMENT-PROGRAM TYPE=*FILE` for selected files. The measured values are output (online) to the monitor and/or written to a measurement values file. The file output can be used to evaluate the measurement values later (offline).

5.4.5 Throughput optimization with AutoDAB (complex I/O load)

A synthetic I/O benchmark with the following access profiles was configured to demonstrate the effects of AutoDAB:

- Access profile: “random”
10 files with pure random accesses and 25% write share
- Access profile: “random local”
10 files with random accesses and good temporal locality with 25% write share
- Access profile: “sequential”
10 files with pure sequential accesses and 50% write share

The data volume involved was twice the cache size (6% sequential or 47% random access with or without temporal locality) - in total 16.8 GB.

This load generates an I/O bottleneck if DAB is not used. Using DAB speeds up the accesses to such a degree that a CPU bottleneck occurs. This shows that the I/O throughput when using DAB is dependent on the available CPU performance.

The measurement was taken on an S server (e.g. with 650 RPF) with the following configuration:

- Symmetrix disk storage system with 8-GB cache (use of 2 pubsets each with 5 disks)
- Global storage with 8 GB (used by one cache area)

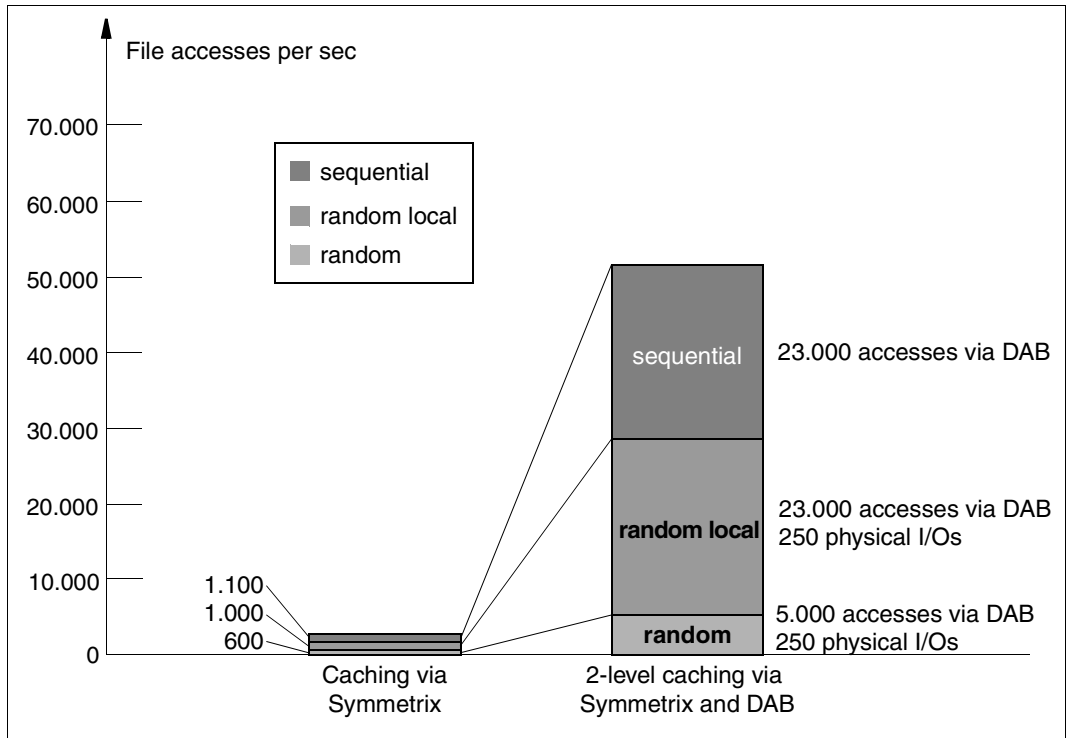
Illustration of the results

Figure 14: Effect of 2-level caching with Symmetrix and AutoDAB on the three access profiles

Explanation of the benchmark

The overall throughput of the benchmark is considerably increased through 2-level caching.

What are the causes of the throughput increase?

The better the hit rate in a DAB cache, the more CPU power can be converted into I/O throughput.

1. The sequentially processed files have a very high hit rate due to the high prefetching factor (data areas are cached in advance) and are kept practically resident in the DAB cache due to the low data volume.
2. The random files with temporal locality also have very high hit rates due to the intelligent algorithms of the AutoDAB and occupy only a small portion of the cache memory.

3. DAB's effective use of the cache memory makes an even greater portion available for the caching of files with random access patterns; this also helps to increase the throughput.

What can be achieved with less GS?

The AutoDAB algorithms stipulate that in the event of a smaller DAB cache, random files without locality will be the first to be “deselected” from the cache, i.e. no longer cached, until all accesses on these files are executed solely on the disk subsystem.

If the cache is reduced even further, random files with temporal locality (sometimes) will also be removed from the cache.

This will keep the increase in throughput at a constantly high level, however the proportions of the relevant access types will shift so that sequential files will be even more preferred, other access types less so.

5.4.6 Throughput increase with DAB with an I/O load distributed on two servers in the XCS

DAB can be used to set up PFA cache areas for shared pubsets in read/write mode. This enables efficient utilization of a shared pubset network as a load-sharing network.

A distribution of a physical disk I/O load cannot bring about an improvement in the throughput or the response time on its own, as all I/Os continue to be processed on the disks as before. When DAB is utilized, however, the CPU power of multiple systems can be used to process the I/O jobs, thus achieving an increase in throughput.

To illustrate this in a quantitative way, the synthetic I/O load described in the section before was distributed and monitored on two lower-performance servers in a parallel HIPLEX (XCS with shared GS).

The load was distributed in such a way that each server had to process half of the data area in comparison with the previous measurement, but the number of processing tasks for each server was retained (in accordance with the multiplied server processing power).

The measurement was taken in a configuration with two S servers (one model with 330 RPF and one model with 175 RPF):

- Symmetrix disk storage system with 8-GB cache (use of two shared pubsets each with 8 disks)
- Global storage with 8 GB (used by two cache areas)

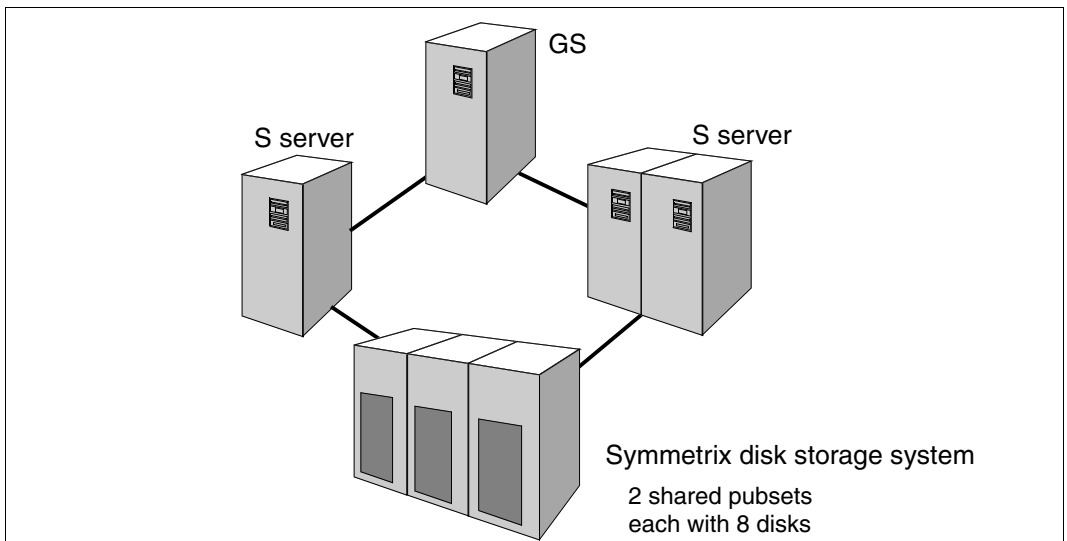


Figure 15: Configuration for the utilization of a shared pubset network in the load-sharing network

The figure below compares the maximum achievable throughput when using the disk subsystem alone with the throughput achieved through the additional utilization of DAB cache areas on both pubsets.

Illustration of the results

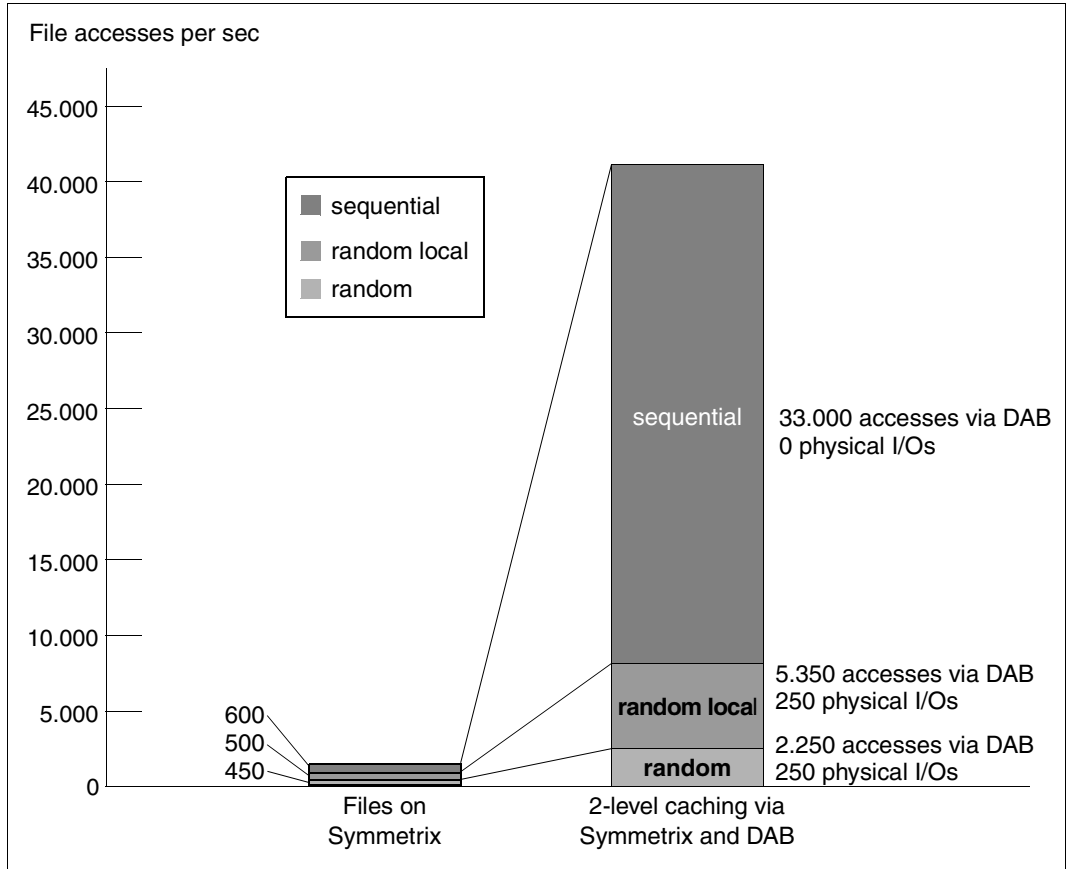


Figure 16: Increased throughput with the utilization of DAB cache areas for shared pubsets

The figure shows the increase in the maximum achievable I/O throughput when using DAB cache areas for shared pubsets, compared with processing via the Symmetrix cache. In addition to the 8 GB Symmetrix cache, a further 8 GB DAB cache in the GS is used.

The following differences can be seen in comparison with measurement on a single system (see [figure 14 on page 75](#)):

- The throughput of the Symmetrix disk storage systems alone was reduced slightly (a total of 1,550 accesses). This is explained by, on the one hand, the doubling of the number of accessing tasks, whereby the sequential files make less of a contribution to a good hit rate for the disk controller cache. On the other hand, the accesses are now made from two servers, which results in synchronization losses in the disk controllers.
- The total throughput when DAB cache areas are used almost reaches the value for the more powerful S server in [section “Throughput optimization with AutoDAB \(complex I/O load\)” on page 74](#). As a result of distributing the load on two servers, DAB can now use the full CPU power of both lower-performance servers (in the section mentioned before), which directly increases the I/O throughput. The major beneficiaries of automatic caching are the sequential files with their extremely high hit rates.

6 Installing, starting and stopping DAB

This chapter provides information about installing the product DAB as well as on starting and stopping the DAB subsystem.

6.1 Installing DAB

DAB is managed as a subsystem by DSSM (Dynamic SubSystem Management).

DAB V9.3 can be only be run under BS2000/OSD-BC V9.0.

The DAB subsystem consists of the following delivery units:

- Subsystem catalog
The subsystem catalog was generated from the subsystem declaration file by SSCM. It contains, among others, references to the LMS module library, the REP file, the SDF syntax file and the message file for DSSM.
- LMS module library
DSSM loads DAB from the LMS module library. The module library contains exactly one link and load module (LLM).
- REP file for object corrections (created from the RMS delivery unit)
- Message file with message and help texts
- SDF syntax file
The SDF syntax file contains the syntax of DAB commands. The commands can only be entered by systems support staff and require the TSOS or SW-MONITOR-ADMINISTRATION privileges (see also the relevant commands in [chapter “Commands” on page 85](#)). The syntax file is activated at subsystem startup as the subsystem syntax file.

- Subsystem initialization file
The subsystem initialization file SYSSSI.DAB.<version> contains the global parameters of the subsystem which mainly influence the caching algorithm and thereby also the performance behavior. These parameters should normally not be modified at all or, in exceptional cases, be modified by Fujitsu Technology Solutions consultants or by specialists for optimizing the cache algorithms.

The sole exceptions are the parameters:

- Parameters for defining the duration of the statistics interval for the statistical data output by /SHOW-DAB-CACHING

The specification of the statistics interval for the cumulative data that is output using /SHOW-DAB-CACHING is carried out using the parameter PERIOD-OF-STATISTICS-INTERVAL. The possible values lie between 1 and 8784 and the unit used is hours. This means that you can set an interval of anything between an hour and a year. The default value for the interval is 24 hours. After the interval has elapsed, the statistics data for the cache area is reset and counting of I/O operations starts at zero again.

With a default value of 24 hours, the resetting of the statistics data is carried out at 00:00 local time, for larger values, the reset is carried out 00:00 of the specified following day (rounded up).

- Parameters for handling access errors on a GS unit for cache areas in a dual partition

The handling of access errors on a GS unit for cache areas in a dual partition is specified using the parameter ACTION-AT-DUAL-GS-ERROR. The possible values are BY-OPERATOR, CONTINUE-GS-ACCESS, CONTINUE-CACHING-MODE-READ, STOP and HOLD-GS-ACCESS. The default value is BY-OPERATOR. The effects of the individual values in the event of the specified error are described in detail in the [section “Detachment/failure of a GS unit with dual data storage” on page 148](#).

- Parameters used to specify the GS cache areas for shutdown and the restoration options for read cache areas.

Controlling the release of cache areas at SHUTDOWN:

- KEEP-GS-CACHE-BUFFER-AFTER-SHUTDOWN = YES:

When carrying out a system shutdown, the ADM-PFA cache areas still available in the GS are not released. These cache areas are then restored the next time the system is started or when the occupancy of a data volume is restored. Steps are taken to ensure that all data in the cache, if necessary, is written back (saved) to the associated data volume.

- KEEP-GS-CACHE-BUFFER-AFTER-SHUTDOWN = NO:
When carrying out a system shutdown, the cache areas still available are released.

Controlling the restoration of ADM-PFA cache areas with CACHING-MODE = READ:

- GS-READ-CACHE-BUFFER-RECOVERY = ENABLED:
Cache areas with CACHING-MODE = READ are also restored when the data volume you are using is allocated for the first time. This parameter setting must be effective in the DAB subsystem parameters both at the time when the cache area is setup as well as at the time when restoration takes place. Otherwise, no restoration of this class of cache area is carried out.
 - GS-READ-CACHE-BUFFER-RECOVERY = DISABLED:
ADM-PFA cache areas with CACHING-MODE = READ cannot be restored.
- Structure and installation file for installation with IMON
This file contains the logical path names of the subsystem files which are supplied with DAB.

Files required for installing DAB (<ver>=093 for DAB V9.3):

SYSLNK.DAB.<ver>	Module library (S servers)
SKMLNK.DAB.<ver>	Module library (SQ servers)
SYSRMS.DAB.<ver>	RMS delivery unit (for creating the REP loader)
SYSREP.DAB.<ver>	REP loader (IMON standard installation created)
SYSSSC.DAB.<ver>	Subsystem catalog
SYSSDF.DAB.<ver>	Syntax file
SYSMES.DAB.<ver>	Message file
SYSSSI.DAB.<ver>	DAB subsystem initialization file
SYSSII.DAB.<ver>	Structure and installation file for installation with IMON

6.2 Starting and stopping the DAB subsystem

The files listed above must be installed in the system prior to starting the DAB subsystem for the first time.

DAB is loaded automatically at BS2000 startup (setting in the supplied standard subsystem catalog entry).

The command `/STOP-SUBSYSTEM SUBSYSTEM-NAME=DAB[,VERSION=09.3]` can be used to unload DAB during the session (i.e. without BS2000 shutdown) under the following conditions:

- No (more) cache areas exist.
- No SM2 measurements are being made for DAB or PFA (active measuring can be stopped by means of the SM2 statement `//STOP-MEASUREMENT-PROGRAM TYPE=*DAB or *PFA`).
- All asynchronous I/O jobs started during cache operation for a cache area released by means of `/STOP-DAB-CACHING` or `/STOP-PUBSET-CACHING` have terminated.
- No action requesting a GS partition is active (in particular no reconstruction of a GS partition during initial allocation of a volume).

DAB can subsequently be (re)started using the command

`/START-SUBSYSTEM SUBSYSTEM-NAME=DAB[,VERSION=09.3A]`.



- If the BS2000 system is terminated despite the fact that it still has cache areas configured (shutdown), the data of these cache areas is written back by DAB within the framework of the shutdown and (except for requests by subsystem parameters) then the cache area is released. It is, however, advisable to release any existing cache areas by means of the appropriate DAB or PFA commands prior to system shutdown so that it is possible to react to errors, if required.
- Subsystem shutdown should be checked with the `/SHOW-SUBSYSTEM-STATUS` command (status NOT CREATED). Messages which reject unloading the subsystem are only output to the console or logged in the CONSLOG file.

7 Commands

See the manual “Commands” [4] for a description of the metasyntax used and the general command return codes.

Overview

Command	Function	Caching method	Page
FORCE-DESTROY-CACHE	Force release of an existing PFA cache area	PFA caching	1
FORCE-STOP-DAB-CACHING	Force release of an existing ADM-PFA DAB cache area	ADM-PFA caching	86
MODIFY-DAB-CACHING	Modify the parameters of a DAB cache area dynamically	ADM-PFA caching and PFA caching	89
MODIFY-DAB-PARAMETERS	Modify the DAB subsystem parameters dynamically, Behavior in the event of a DUAL-GS error	ADM-PFA caching and PFA caching	100
SHOW-CACHE-CONFIGURATION	Display the PFA cache area configuration	PFA caching	1
SHOW-DAB-CACHING	Display information about the current DAB configuration	ADM-PFA caching and PFA caching	102
START-DAB-CACHING	Create ADM-PFA DAB cache areas	ADM-PFA caching	123
STOP-DAB-CACHING	Release ADM-PFA DAB cache areas	ADM-PFA caching	139

¹ This command is described in detail in the “Commands” manual [4].

FORCE-STOP-DAB-CACHING

Force release of existing ADM-PFA DAB cache area

Domain: SYSTEM-TUNING

Privileges: TSOS

Function

The command is intended for use only in exceptional circumstances (such as disk defects).

The `/FORCE-STOP-DAB-CACHING` command enables systems support to force ADM-PFA DAB cache areas to be released without writing the data in the cache to disk. All storage areas allocated when the cache area was set up are released, and any data not yet saved to disk is lost. The disk data may subsequently be in an inconsistent state.

Message NDB0045 is output prior to command execution as a warning, indicating which files or volumes may be destroyed after command execution. This is followed by message NDB0046, requesting explicit confirmation of the action before it is actually carried out.

USER-PFA cache areas cannot be released via the `/FORCE-STOP-DAB-CACHING` command.

Format

FORCE-STOP-DAB-CACHING

CACHE-ID = <name 1..32>

,CACHE-MEDIUM = *MAIN-MEMORY / *GS(...)

*GS(...)

| **PARTITION-ID** = <name 1..8>

Operands

CACHE-ID = <name 1..32>

Specifies the DAB cache area which is to be forcibly released.

CACHE-MEDIUM = *MAIN-MEMORY / *GS (...)

The DAB cache area specified with the CACHE-ID operand will not be released unless it is found in the cache medium specified with this operand.

CACHE-MEDIUM = *MAIN-MEMORY

The DAB cache area specified with the CACHE-ID operand will not be released unless it is a main memory cache area.

CACHE-MEDIUM = *GS (...)

PARTITION-ID = <name 1..8>

The DAB cache area specified with the CACHE-ID operand will not be released unless it is in the partition of global storage identified with this operand.



After a reconstruction of GS cache areas from another system, there may be multiple ADM-PFA cache areas with the same cache ID. These are distinguished by the name of the GS partition in which they are embedded. Specifying CACHE-MEDIUM=*GS(PARTITION-ID=...) permits the cache area to be addressed unequivocally.

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	32	CMD0221	Internal SDF error
	64	CMD0216	Required privilege missing
	65	CMD2241	Subsystem not loaded
	64	NDB0005	No authorization to invoke the command
	64	NDB0010	Wrong syntax file version
	64	NDB0019	Specified cache area not found in specified medium
	64	NDB0023	Cache area could not be released
	64	NDB0067	Cache area not released in cache medium due to a system error

Example

```

/show-dab-caching cache-id=dblog
CACHE-ID      = DBLOG
CACHE-MEDIUM = GS (GS-UNIT=1)
PARTITION-ID = DAB12BV (STATE = OPERABLE)          FREE-SIZE =    19 MB
CACHE-SIZE    = 384 KB (BY-FILE)                   SEGMENT-SIZE =   32 KB
CACHING-MODE = READ-WRITE(FORCE-OUT=NO)

```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:   <date> <time>
-----
```

FILE/VSN	AREA	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
=====	=====	=====	=====	=====	=====	=====
:B300:\$DB.SESDB.0001.C.0001	1-192	Y	85737	100%	38474	100%

```

/stop-dab-caching cache-id=dblog
NDB0029 DAB CACHE BUFFER 'DBLOG' IN CACHE MEDIUM 'GS(DAB12BV)' WAS NOT
COMPLETELY DESTAGED TO DISK STORAGE DUE TO ERROR
NDB0023 DAB CACHE BUFFER 'DBLOG' IN CACHE MEDIUM 'GS(DAB12BV)' WAS NOT
DELETED

```

```

/show-dab-caching cache-id=dblog
CACHE-ID      = DBLOG
CACHE-MEDIUM = GS (GS-UNIT=1)
PARTITION-ID = DAB12BV (STATE = OPERABLE)          FREE-SIZE =    19 MB
CACHE-SIZE    = 384 KB (BY-FILE)                   SEGMENT-SIZE =   32 KB
CACHING-MODE = READ-WRITE(FORCE-OUT=NO)

```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:   <date> <time>
-----
```

FILE/VSN	AREA	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
=====	=====	=====	=====	=====	=====	=====
:B300:\$DB.SESDB.0001.C.0001	1-192	Y(PINNED-DATA)	85737	100%	38474	100%

```

/copy-file :b300:$db.sesdb.0001.c.0001,:b301:$sv.sesb.save-file
/force-stop-dab-caching cache-id=dblog,cache-medium=*gs(partition-id=dab12bv)
NDB0045 THE FOLLOWING FILE(S) MAY BE DESTROYED BY THE EXECUTION OF A FORCE-
STOP-DAB COMMAND FOR THE CACHE BUFFER 'DBLOG': :B300:$DB.SESDB.0001.C.0001
NDB0046 AFTER THE EXECUTION OF THE FORCE-STOP-DAB COMMAND THE DATA AREA
SERVICED BY THE CACHE BUFFER 'DBLOG' MAY BE DESTROYED. SHOULD THIS COMMAND BE
EXECUTED? REPLY (Y=YES ; N=NO)?

```

```

y
NDB0022 DAB CACHE BUFFER 'DBLOG' IN CACHE MEDIUM 'GS(DAB12BV)' HAS BEEN
DELETED

```


MODIFY-DAB-CACHING

Modify parameters of DAB cache area dynamically

Domain: SYSTEM-TUNING

Privileges: TSOS

Function

The /MODIFY-DAB-CACHING command allows systems support to modify all of the main parameters of a DAB cache area dynamically.

The modified parameters include the cache size and the cache segment size, along with the data of file-specific cache areas (created with the command /START-DAB-CACHING AREA=*FILE/*BY-SYSTEM), the caching mode and the FORCE-OUT parameter.

The command can be used for both ADM-PFA and (user) PFA cache areas.

Format

MODIFY-DAB-CACHING	Alias: MDDABC
<p>CACHE-ID = <alphanum-name 1..32></p> <p>,CACHE-SIZE = <u>*UNCHANGED</u> / *EXTEND (...) / *REDUCE (...)</p> <p> *EXTEND(...)</p> <p> NEW-SIZE = <integer 1..8388608>(…)</p> <p> <integer 1..8388608>(…)</p> <p> DIMENSION = *KILOBYTE / *MEGABYTE</p> <p> *REDUCE(...)</p> <p> NEW-SIZE = <integer 0..8388608>(…)</p> <p> <integer 0..8388608>(…)</p> <p> DIMENSION = *KILOBYTE / *MEGABYTE</p> <p>,CACHE-SEGMENT-SIZE = <u>*UNCHANGED</u> / *32 / *16 / *8 / *4</p> <p>,AREA = <u>*UNCHANGED</u> / *ADD-FILE(...) / *REMOVE-FILE(...) / *RESET-FILE(...) /</p> <p> *ADD-PUBSET(...) / *REMOVE-PUBSET(...)</p> <p> *ADD-FILE(...)</p> <p> FILE-NAME = <filename 1..54 without-vers> / <partial-filename 2..53></p> <p> *REMOVE-FILE(...)</p> <p> FILE-NAME = <filename 1..54 without-vers> / <partial-filename 2..53></p> <p> *RESET-FILE(...)</p> <p> FILE-NAME = <filename 1..54 without-vers> / <partial-filename 2..53></p> <p> *ADD-PUBSET(...)</p> <p> PUBSET = <cat-id 1..4></p> <p> *REMOVE-PUBSET(...)</p> <p> PUBSET = <cat-id 1..4></p> <p>,CACHING-MODE = <u>*UNCHANGED</u> / *READ / *READ-WRITE / *WRITE / *BY-CACHE-MEDIUM</p> <p>,FORCE-OUT = <u>*UNCHANGED</u> / *AT-LOW-FILLING / *AT-HIGH-FILLING / *NO</p>	

Operands

CACHE-ID = <alphanum-name 1..32>

Identifier of the cache area to be modified.

CACHE-SIZE = *UNCHANGED / *EXTEND(...) / *REDUCE(...)

Defines the size of the cache area.

The cache size of user PFA cache areas for shared pubsets cannot be changed dynamically.

CACHE-SIZE = *UNCHANGED

The size of the cache area is not to be changed.

CACHE-SIZE = *EXTEND(...)

The cache area should be extended.



For cache areas in the GS with a GS partition > 1 GB the new size of the cache area should be specified in multiples of 1 MB, or otherwise in multiples of 32 KB. When necessary, DAB will round the figure up to the next corresponding multiple.

NEW-SIZE = <integer 1..8388608>(…)

Specifies the new size of the cache area.

The process of extending the cache area to the specified size is additive, all cache data already imported is retained.

In the case of ADM-PFA cache areas in the main memory with MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE (see /START-DAB-CACHING on [page 123](#)), the size of the cache area set (desired) is increased. The current size is increased when the command is executed if the memory utilization permits this.

DIMENSION = *KILOBYTE / *MEGABYTE

Defines whether the cache area size is to be specified in KB or in MB.



The more memory used in the main memory cache medium for caching, the less memory there is available for paging activities. A rise in the paging rate reduces the performance gain for DAB and should therefore be avoided.

CACHE-SIZE = *REDUCE(...)

The cache area is to be reduced.

NEW-SIZE = <integer 0..8388608>(…)

Specifies the new size of the cache area.

The following applies for cache areas with CACHING-MODE=*WRITE/*READ-WRITE:
Only cache data that is needed is saved to disk, thus helping reduce the cache area to the required size. As for all types of caching mode, the cache segments of the cache area are then disconnected. Caching in all of the other cache segments is not affected by this action.

In the case of ADM-PFA cache areas in the main memory with MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE (see /START-DAB-CACHING on [page 123](#)), the size of the cache area set (desired) is decreased. The current size is decreased only if the newly set size is below the current size.

DIMENSION = *KILOBYTE / *MEGABYTE

Defines whether the cache area size is to be specified in KB or in MB.

CACHE-SEGMENT-SIZE = *UNCHANGED / *32 / *16 / *8 / *4

Specifies which cache segment size should be set.

Changing the previous cache segment size has no effect on cache areas with automatic caching (AutoDAB).

CACHE-SEGMENT-SIZE = *UNCHANGED

The cache segment size is not to be changed.

CACHE-SEGMENT-SIZE = *32 / *16 / *8 / *4

The cache segment size is to be changed to 32KB / 16KB / 8KB / 4KB.

AREA = *UNCHANGED / *ADD-FILE(...)/ *REMOVE-FILE(...)/ *RESET-FILE(...)/ *ADD-PUBSET(...)/ *REMOVE-PUBSET(...)

The data area of an ADM-PFA file-specific cache area is to be changed.

- In the case of a file-specific cache area (created with AREA=*FILE), individual files can be added to or removed from the cache.
- In the case of an automatic cache area (created with AREA=*BY-SYSTEM), one or more files can be permanently added to or removed from the cache. It can also be specified that these files should be cached automatically again, i.e. depending on what AutoDAB decides.
- In the case of an automatic cache area (configured with AREA=*BY-SYSTEM), a pubset can be permanently added to the cache or removed from it.

AREA = *UNCHANGED

The data area of the cache area is not to be changed.

AREA = *ADD-FILE(...)

The data area of the cache area is to be extended by one or more files. With AutoDAB, this file is cached without first being assessed by DAB.

FILE-NAME = <filename 1..54 without-vers>

Name of the file to be added to the specified cache area.

FILE-NAME = <partial-filename 2..53>

*Only for automatic cache areas (created with AREA=*BY-SYSTEM):*

Partially qualified file name for a file set which is to be added to the specified cache area.

AREA = *REMOVE-FILE(...)

The data area of the cache area is to be reduced by one or more files. The cache data is written to the disk(s) and invalidated in the cache.

FILE-NAME = <filename 1..54 without-vers>

Name of the file to be removed from the specified cache area.

FILE-NAME = <partial-filename 2..53>

*Only for automatic cache areas (created with AREA=*BY-SYSTEM):*

Partially qualified file name for a file set which is to be removed from the specified cache area.

AREA = *RESET-FILE(...)

*Only for automatic cache areas (created with AREA=*BY-SYSTEM):*

The data area of the automatic cache area is to be modified in such a manner for one or more files that these files are once more automatically cached, i.e. depending on what AutoDAB decides.

FILE-NAME = <filename 1..54 without-vers> / <partial-filename 2..53>

Name of the file or partially qualified file name for a file set which is once more to be cached automatically in the specified cache area.

AREA = *ADD-PUBSET(...)

*Only for automatic cache areas (created with AREA=*BY-SYSTEM):*

The database of the cache area is to be extended by one pubset.

PUBSET = <cat-id 1..4>

Name of the pubset which is to be added to the specified cache area.

AREA = *REMOVE-PUBSET(...)

*Only for automatic cache areas (created with AREA=*BY-SYSTEM):*

The database of the cache area is to be reduced by one pubset.

PUBSET = <cat-id 1..4>

Name of the pubset which is to be removed from the specified cache area.

CACHING-MODE = *UNCHANGED / *READ / *READ-WRITE / *WRITE / *BY-CACHE-MEDIUM

The caching mode of the cache area is to be changed. This is only possible for ADM-PFA cache areas.

There are two different “directions” to be considered when changing the caching mode:

- When changing from *READ to *READ-WRITE/*WRITE/*BY-CACHE-MEDIUM, the imported read data is retained; in the future write data will also (or only in the case of *WRITE) be imported in the cache.
- In the other direction (to *READ), data is written to the disk(s) without disk copy, only read data is then imported in the cache.

CACHING-MODE = *UNCHANGED

The caching mode is not changed.

CACHING-MODE = *READ

The cache area is to be changed to a read cache. Write data imported in the cache must be written to the disk(s).

CACHING-MODE = *READ-WRITE

The cache area is to be changed to a read/write cache.

CACHING-MODE = *WRITE

The cache area is to be changed to a write cache.

CACHING-MODE = *BY-CACHE-MEDIUM

The caching mode set is to depend on the cache medium and file attribute (see the /START-DAB-CACHING command).

FORCE-OUT = *UNCHANGED / *AT-LOW-FILLING / *AT-HIGH-FILLING / *NO

The threshold for the asynchronous saving of cache data is to be changed. This determines if and when write data is to be written from the cache to the disk(s) (see the /START-DAB-CACHING command).

FORCE-OUT = *UNCHANGED

The threshold for the asynchronous saving of cache data is not to be changed.

FORCE-OUT = *AT-LOW-FILLING

The number of cache segments not saved to disk is to be kept to a minimum through threshold-controlled save runs. Saving is to be triggered as soon as 25% of the cache is filled with unsaved write data.

FORCE-OUT = *AT-HIGH-FILLING

For the cache segments of this cache area, only as many data savings as are required are to be carried out in order to have enough segments available for new cache imports. Saving is to be triggered as soon as 75% of the cache is filled with unsaved write data.

FORCE-OUT = *NO

The data of this cache area is not to be transferred to the disk in threshold-controlled save runs. The data is only transferred to the disk when the cache area is disconnected using the /STOP-DAB-CACHING command.

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	32	CMD0221	Internal SDF error
	64	CMD0216	Required privilege missing
	64	NDB0002	Pubset with key disks cannot be added to the cache area without keys
	64	NDB0005	No authorization to invoke command
	64	NDB0006	Pubset cannot be added to file area cache
	64	NDB0010	Wrong syntax file version
	64	NDB0012	DMS error
	64	NDB0026	File is not cataloged
	64	NDB0027	File cannot be served because it is either a tape file, has been migrated by HSMS, is a file group, a work file for SO migration or a work file for SPACEOPT
	64	NDB0028	File already served
	64	NDB0029	Cache area could not be fully saved to disk
	64	NDB0032	Data areas on a user PFA pubset which are served by DAB cannot be served in ADM-PFA cache areas
	64	NDB0034	File on key disk cannot be added to the cache area without keys
	64	NDB0055	Data on GS volumes not served
	64	NDB0061	Data of volume cannot be served in two different GS partitions
	64	NDB0071	No write caching is possible for the specified file because of file encryption
	64	NDB0072	User ID of the specified file is unknown
	64	NDB0080	Caching of home pubset data is not permissible in a write or read/write cache
	64	NDB0081	Caching of home pubset data in parallel HIPLEX using cache medium GS in conjunction with a write or read/write cache is not permissible
	64	NDB0096	The modification could not be performed either because of a memory bottleneck, a GS access error, a bottleneck in the GS, a malfunction in cache destaging or an error in a function in BS2000
	64	NDB0097	The cache area is unknown, exists more than once or has been reconfigured
	64	NDB0098	Pubset not imported locally
	64	NDB0099	Pubset from GS volumes is not served
	64	NDB0100	Pubset already being served

(SC2)	SC1	Maincode	Meaning
	64	NDB0101	More than one SM pubset specified for read/write caching
	64	NDB0103	File catalog cannot be served (error)
	64	NDB0110	SM pubset does not contain a cacheable volume set
	64	NDB0111	An ADD-FILE or REMOVE-FILE is not permissible for the cache area
	64	NDB0112	The file is not located on an occupied disk volume
	64	NDB0113	The cache size of cache areas with a dynamic cache size cannot be modified explicitly
	64	NDB0114	Parameter error during extension: new size is smaller than or equal to the old size
	64	NDB0115	Parameter error during reduction: new size is larger than or equal to the old size
	64	NDB0116	Modification of the cache segment size is not possible with automatic caching
	64	NDB0117	The size of a user PFA cache area for a shared pubset cannot be modified
	64	NDB0118	Modification of the cache area to caching mode *WRITE/*READ-WRITE is not possible because files are being served on the home pubset
	64	NDB0119	The command cannot be executed at present because another MODIFY-DAB-CACHING command is already being executed
	64	NDB0120	Parameter error in the MODIFY-DAB-CACHING command: the file(s) is(are) not served from the specified cache area
	64	NDB0121	A switchover of the caching mode is not permitted for a user PFA cache area
	64	NDB0123	Write caching cannot be configured in the global storage as other pubsets in addition to an SM pubset are being cached in the cache area
	64	NDB0124	The new cache segment size is the same as the old cache segment size
	64	NDB0125	The cache size or the cache segment size cannot be modified because of a preceding GS access error
	64	NDB0126	GS partition is reserved for PFA cache areas of shared pubsets
	64	NDB0127	The file cannot be served in the specified cache area
	64	NDB0129	Modification of the cache segment size of a cache area without cache memory is not possible
	64	NDB0155	Partial file names are not supported as no cache area with automatic caching exists
	64	NDB0156	The RESET-FILE parameter is not supported for the specified cache area
	64	NDB0158	The cache area has been reduced because of the lack of main memory or storage space in the global storage
	64	NDB0159	File cannot be added to the cache area owing to lack of main memory or space in the global storage

(SC2)	SC1	Maincode	Meaning
	64	NDB0160	SM pubset cannot be inserted in a write cache in the partition on the global storage
	64	NDB0161	Pubset cannot be removed from the cache area owing to pubset import
	64	NDB0162	Processing of a pubset was aborted when a particular volume was accessed
	64	NDB0163	Processing of a pubset was aborted when a volume was accessed
	64	NDB0165	Processing of a pubset was aborted when a particular volume was accessed. It is not known how processing was concluded
	64	NDB0166	Processing of a pubset was aborted when a volume was accessed. It is not known how processing was concluded
	64	NDB0168	Parameter error: pubset to be removed is not served by DAB in the cache area
	64	NDB0169	The maximum number of pubsets is already being served in the cache area
	64	NDB0170	Parameter error: pubset to be added is not served by DAB in the cache area
	64	NDB0177	The configuration of a cache area cannot be modified during main memory reconfiguration
	64	NDB0197	Pubset cannot be added to the cache area owing to lack of main memory or space in the global storage
	65	CMD2241	DAB subsystem not loaded

Notes

1. Handling files which correspond to the specification of different partially qualified file names:

Specifying partially qualified file names in the `AREA=*ADD-FILE(...)/REMOVE-FILE(...)/RESET-FILE(...)` operand enables subsets of files to be created which differ from a specific superset of files.

If, for example, the files which begin with the partial name `:<catid>:$TSOS.SYS.` are not to be added to the automatic cache area, this is done using:

```
/MODIFY-DAB-CACHING CACHE-ID=<cache-id>,AREA=*REMOVE-FILE(SYS.)
```

On the other hand, files which begin with `:<catid>:$TSOS.SYS.SDF.` should always be cached. The following command ensures that these files are always cached:

```
/MODIFY-DAB-CACHING CACHE-ID=<cache-id>,AREA=*ADD-FILE(SYS.SDF.)
```

In this case files which begin with `:<catid>:$TSOS.SYS.<xyz>.` where `xyz <> SDF`, are not cached. However, the commands must be issued in the specified order. If they were issued in reverse order, none of the files with `:<catid>:$TSOS.SYS.SDF.` would be cached.

Specifying a shorter prefix cancels all preceding specifications for longer, partially qualified or complete file names which begin with a prefix.

2. Renaming files:

When a file which was added to the data area of the cache area with `ADD-FILE` or removed from it with `REMOVE-FILE` is renamed, it retains this attribute even after it has been renamed.

When a file which was added to the data area of the cache area with `ADD-FILE` or removed from it with `REMOVE-FILE` by specifying a partially qualified file name is renamed, it retains this attribute even after it has been renamed.

When a file which was added to the data area of the cache area with `ADD-FILE` or removed from it with `REMOVE-FILE` by specifying a partially qualified file name is renamed (in these cases the file loses this attribute), or if neither `ADD-FILE` nor `REMOVE-FILE` has been applied to it, it takes over the attribute regarding affiliation to the data area which applies for the new file name. If the file is given a new name which matches the previous specification for a partially qualified file name, it is added to the data area provided this partially qualified file name was specified in `ADD-FILE`. The same applies for removing files if the partially qualified file name was specified in `REMOVE-FILE`. Otherwise DAB decides on whether the file should be cached.

3. Modifying the cache segment size

When the cache segment size is modified in a user PFA cache or a file area cache, it can occur that the old cache size cannot be set again and the cache is continued with a reported smaller cache size.

4. Contending cache areas

The current size of a cache area in the main memory (MEMORY =*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE/*ANY) can be too large because it was configured first. Another cache area which was configured later using this MEMORY setting is consequently too small. This is a case in which the current sizes of the cache areas should be modified.

To do this, the size set for the larger cache area must be decreased. After this has been done and a wait time of approx. two minutes, the current size of the smaller cache area should have increased (see the /SHOW-DAB-CACHING command). If the increase is not sufficient, the size of the larger cache area must be reduced again. If the increase is sufficient, the size of the larger cache area can be reset to its original value.

MODIFY-DAB-PARAMETERS

Modify the subsystem parameters dynamically

Domain: System tuning

Privileges: TSOS

Function

The command /MODIFY-DAB-PARAMETERS can be used to dynamically modify selected subsystem parameters of DAB. The modification of the subsystem parameters is only effective until the next time a change is made, or until the end of the current DAB session (unloading the DAB subsystem).

Format

MODIFY-DAB-PARAMETERS	Alias: MDDABP
ON-DUAL-GS-ERROR =*UNCHANGED / *STD / *CONTINUE-GS-ACCESS / *CONTINUE-CACHING-MODE-READ / *HOLD-GS-ACCESS / *BY-OPERATOR	

Operands

ON-DUAL-GS-ERROR= *UNCHANGED / *STD / *CONTINUE-GS-ACCESS / *CONTINUE-CACHING-MODE-READ / *HOLD-GS-ACCESS / *BY-OPERATOR

Specifies the way DAB is to react to dual GS access errors.

ON-DUAL-GS-ERROR = *UNCHANGED

Does not modify the way DAB reacts to dual GS access errors.

ON-DUAL-GS-ERROR = *STD

DAB reacts to dual GS access errors in the way specified in the subsystem parameter file.

ON-DUAL-GS-ERROR = *CONTINUE-GS-ACCESS

If it is possible to access at least one GS unit of a dual partition, then the access is considered to have been carried out successfully (mono operation).

ON-DUAL-GS-ERROR = *HOLD-GS-ACCESS

If a dual GS error occurs, access to the storage is considered not to have been carried out successfully. Data transfer requests are terminated by displaying a I/O error (switch to domino mode).

ON-DUAL-GS-ERROR = *CONTINUE-CACHING-MODE-READ

If a dual access error occurs, the system writes back the cache data for all affected cache areas and then switches the cache areas to read caching. Once the cause of the error has been eliminated, the system returns to the original caching mode.

ON-DUAL-GS-ERROR = *BY-OPERATOR

If a dual access error occurs, the system administrator is asked how to react to the error for each of the affected cache areas.

The possible responses are (see also [page 148](#)):

- C = CONTINUE-GS-ACCESS (mono mode)
- R = CONTINUE-CACHING-MODE-READ (read caching)
- S = STOP (release cache area)

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	32	NDB0005	No authorization to invoke the command
	64	CMD0216	Required privilege missing
	64	NDB0010	Wrong syntax file version
	64	NDB0153	System error when invoking command
	65	CMD2241	Subsystem not available

SHOW-DAB-CACHING

Display information on current DAB configuration

Domain: SYSTEM-TUNING

Privileges: TSOS
SW-MONITOR-ADMINISTRATION

Function

The /SHOW-DAB-CACHING command supplies systems support staff with information about the cache areas currently installed in the cache media main memory and global storage. The cache areas about which information is to be displayed can be selected either via the cache ID or via the cache medium. The information displayed can include either all status data, a summary of statistic data or (with AREA=*BY-SYSTEM) the currently served and unserved data areas for each cache area.

The information can be requested for ADM-PFA or PFA cache areas.



The values supplied by openSM2 refer to the most recent time interval and thus usually differ from the cumulated statistics displayed by /SHOW-DAB-CACHING.

The command supports structured output in S variables (see [page 108](#)).

Format

SHOW-DAB-CACHING	Alias: SHDABC
<p>CACHE-ID = *ALL / <alphanum-name 1..32></p> <p>,CACHE-MEDIUM = *ALL / *MAIN-MEMORY / *GS(...)</p> <p> *GS(...)</p> <p> PARTITION-ID = *ALL / <name 1..8></p> <p>,INFORMATION = *STD / *SUMMARY / *SYSTEM-CACHED-FILES(...)</p> <p> *SYSTEM-CACHED-FILES(...)</p> <p> CACHING = *ANY / *ACTIVE / *SUSPENDED</p>	

Operands

CACHE-ID = *ALL / <alphanum-name 1..32>

Identifies the DAB cache area about which information is to be displayed.

CACHE-ID = *ALL

The configuration and statistical data of all existing cache areas is to be displayed.

CACHE-ID = <alphanum-name 1..32>

Name of the cache area for which configuration and statistical data is to be displayed.

CACHE-MEDIUM = *ALL / *MAIN-MEMORY / *GS(...)

Specifies the cache medium in which the cache area is located.

CACHE-MEDIUM = *ALL

Information about the cache area identified by the CACHE-ID operand is to be displayed, irrespective of the cache medium it is located in.

CACHE-MEDIUM = *MAIN-MEMORY

Information about the cache area identified by the CACHE-ID operand will be displayed, provided the cache area is located in main memory.

CACHE-MEDIUM = *GS(...)

Information about the cache area identified by the CACHE-ID operand will be displayed, provided the cache area is located in global storage.

PARTITION-ID = *ALL / <name 1..8>

Specifies the DAB partition in global storage which is to be searched for the DAB cache area identified by the CACHE-ID operand.

PARTITION-ID = *ALL

Information about the DAB cache area identified by the CACHE-ID operand is to be displayed irrespective of the DAB partition in global storage it is found in.

PARTITION-ID = <name 1..8>

Restricts the search for the DAB cache area identified by the CACHE-ID operand to a specific DAB partition in global storage.



After a reconstruction of GS cache areas from another system, there may be multiple ADM-PFA cache areas with the same cache ID. These are distinguished by the name of the GS partition in which they are embedded. Specifying CACHE-MEDIUM=*GS(PARTITION-ID=...) permits the cache area to be addressed unequivocally.

Specifying PARTITION-ID = *ALL causes information about all cache areas with the specified cache ID in all GS partitions to be displayed.

INFORMATION = *STD / *SUMMARY / *SYSTEM-CACHED-FILES(...)

Determines the scope of information output.

INFORMATION = *STD

All information available for the cache area is to be output, including the volume/file list.

INFORMATION = *SUMMARY

Only global information about the cache area is to be output (no volume/file list but cumulated statistics).

INFORMATION = *SYSTEM-CACHED-FILES(...)

The selection of files for which information is to be displayed can be restricted for cache areas using automatic caching.

CACHING = *ANY / *ACTIVE / *SUSPENDED

Defines the selection of files for which information is to be output.

CACHING = *ANY

All files using DAB are included.

CACHING = *ACTIVE

Information is output about the files being currently served.

CACHING = *SUSPENDED

Information is output about all files whose caching has been (temporarily) stopped.

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	32	CMD0221	Internal SDF error
	64	CMD0216	Required privilege missing
	64	CMD2009	Error while generating the S variables
	64	NDB0005	No authorization to invoke the command
	64	NDB0010	Wrong syntax file version
	64	NDB0019	Specified cache area not found in specified medium
	64	NDB0024	No cache area currently active
	64	NDB0068	Cache area not found
	64	NDB0069	No cache area in the specified medium is currently active
	64	NDB0079	Command not executed due to a system error
	65	CMD2241	Subsystem not available

Format of information output

The following information is output for each selected cache area:

- Name of the cache area (cache ID)
For ADM-PFA cache areas, the name can be freely selected for the /START-DAB-CACHING command (DEFAULT=BUFFER#nnn), for PFA cache areas the name corresponds to the pubset ID (SF pubset) or the volume set ID (SM pubset).
- Cache medium of the cache area
Cache medium used by this cache area: main memory (M-M) / global storage (GS).
- Global storage-specific data
 - GS unit specification (1, 2 or DUAL)
GS unit 1, 2 or DUAL, specifies the GS unit used by this cache area and the data storage mode of the cache area.
 - GS partition ID
Used partition of this cache area.
 - Status of GS partition:
 - OPERABLE: Partition is available
 - MONO: An error was detected on a GS unit when accessing a dual partition; caching is continued on a GS unit.
 - DEFECT: An error was detected when accessing the cache medium; reading of the cache data may no longer be possible
- Free space in the current partition (with cache media GS)
- Specified (desired) size of the cache area (in the case of the cache media main memory and GS)
This can be varied in the case of ADM-PFA cache areas by specifying AREA=*BY-FILE, which is flagged using the suffix "(BY-FILE)".
- Current size of the cache area (in the case of the cache medium main memory and MEMORY=*ANY/*ABOVE-MIN-MEM-SIZE/*BELOW-MIN-MEM-SIZE)
In the case of ADM-PFA cache areas this can differ from the specified (desired) value because of the specification MEMORY=*ANY/ *ABOVE-MIN-MEM-SIZE. This is flagged using the suffix "(ANY)", "(ABOVE)" or "(BELOW)".
- Segment size of the cache area (not with AutoDAB)

- Caching mode (READ, READ-WRITE, WRITE, USER-PFA or BY-CACHE-MEDIUM);
With caching modes other than READ, additional information about the cache occupancy level which will cause write data of the cache area to be saved to an external volume (FORCE-OUT operand).
If a GS unit in cache areas with GS-UNIT=*DUAL and the error handling setting “READ” (see [chapter “Error handling” on page 145](#)) fails or is disconnected, the suffix “(READ-ONLY)” will be output following the changeover to read mode.

When a cache area used with any of the caching modes mentioned above is released, information about the percentage of write data transferred to the associated external volumes (DESTAGED) is output.

- Support for shared disks (with AREA=*FILE)
(definition whether read caching of a data area is to be performed even if the associated disk is operated in shared private disk or shared pubset mode at initial allocation; write caching will not be performed for the data area in this case).
The information is output only in conjunction with SHARED-DISK-SUPPORT=*YES.
- Start of the current monitoring interval with date/time.
- If applicable, the end of the monitoring interval with date/time if statistical values have overflowed.

With INFORMATION=*STD or =*SYSTEM-CACHED-FILES(...), the list of data areas served is additionally output:

- For cache areas with automatic caching, first the list of specified volumes (pubsets/private disks) with catalog ID or VSN
- File name (AREA=*FILE/*BY-SYSTEM or PFA)
- Data area served (file: size in LHP)
- With INFORMATION=*SYSTEM-CACHED-FILES(CACHING=*ANY), additionally the amount of cache occupied by this file (CACHE-USE)

- Operating state of the data area served (SERVICE)
 - information as to whether any areas are currently being served by DAB (Y) or not (N)
 - specification as to whether the disk is shared Y(SHARED) or not (N(SHARED))
 - indication that the cache area is being released (Y(STOPPED))
 - indication that a file is being served and is encrypted (Y(ENCRYPTED))

Example

```
% :CN31:$TSOS.FALAST.2.FILE.004.3ALE
:%          1-129      Y(ENCRYPTED)    7068868    100%    2512446    0%
% :CN31:$TSOS.FALAST.2.FILE.007.3ALH
%          1-129      Y(ENCRYPTED)    4915150    100%    2662941    0%
```

For files:

- indication that the file / partial qualification is being served (Y)
 - indication whether the file is not being served (N) or is not being served and is closed (N(CLOSED))
 - indication whether a shared file is being served (Y(SHARED)) or not being served (N(SHARED))
 - indication of partial buffering of the file (Y(PARTIAL)) as a result of the file being distributed over several disks, not all of which are allocated exclusively
 - cache data of the data area could not be saved the last time the data area was saved due to a hardware or software error (Y(PINNED-DATA) or N(PINNED-DATA))
 - data description of DAB does not match data description of disk space allocator due to an error (N(INVALID))
 - indication that the file / partial qualification has been explicitly removed (N(REMOVED))
 - indication that the file / partial qualification is being served explicitly (Y(MANDATORY))
 - indication that the encrypted file is being served (Y(ENCRYPTED)) or not being served (N(ENCRYPTED)). Encrypted files are not write-cached.
 - indication that the encrypted file is being explicitly cached (Y(MAND/CRYPT)). Write caching also takes place for this file.
 - indication that the encrypted file is not being served because it is closed (N(CLOSED/CRYPT)).
 - indication that the encrypted and shared file is being served (Y(SHARED/CRYPT)), but it is not write-cached.
 - indication that an encrypted and shared file is being explicitly cached (Y(SH/MAND/CRYPT)), but it is not write-cached.
- Statistics about read and write accesses to the data area and the associated hit rates.

With INFORMATION=*SUMMARY, statistics about read and write accesses to the data area and the associated hit rates are additionally output.

For cache areas with automatic caching, the statistics are split according to the currently buffered files (CACHED). These are classified into sequentially processed (SEQUENTIAL), randomly processed (RANDOM) and undefined access method (UNDEFINED), the files excluded from caching (UNCACHED) and the files which are already closed (CLOSED).



If the cache area to be output is a user PFA cache area on a shared pubset, multiple cache areas with the same name may exist temporarily on the system of the pubset master after reconfiguration in the parallel HIPLEX. These cache areas are output separately (followed by a file list for INFORMATION=*ALL). The additional cache areas available are released once the cache data is saved.

Output in S variables

The S variables form a structure with elements for each cache area. The substructures for the file/volume lists or the statistical data are generated for each cache area.

The INFORMATION operand of the /SHOW-DAB-CACHING command defines which S variables are assigned values. The following values are possible for INFORMATION:

Value of the INFORMATION operand	Abbreviation in table
INFORMATION = *STD / *SYSTEM-CACHED-FILES	1
INFORMATION = *SYSTEM-CACHED-FILES	2
INFORMATION = *SUMMARY	3

Additional conditions which define the assignment of values to S variables:

Additional conditions	Abbreviation in table
var(*LIST).AREA = *FILE / *BY-SYSTEM	a
var(*LIST).AREA = *FILE / *USER-PFA / *BY-SYSTEM	b
var(*LIST).AREA = *FILE / *USER-PFA	c
var(*LIST).AREA = *BY-SYSTEM	d
var(*LIST).CACHE-MED = *GS	e
var(*LIST).GS-UNIT = *DUAL, failure of a GS unit and error handling setting READ	f

Output information	Name of the S variable	T	Contents	Condition
AREA specification: ADM-PFA cache area (AREA=*FILE /*BY-SYSTEM) or User PFA cache area (CACHED-FILE=*BY-SYSTEM- SELECTED / other) *FILE=output is on a file-specific basis *USER-PFA=output is on a file- specific basis *BY-SYSTEM=output is on a file- specific basis	var(*LIST).AREA-TYPE	S	*FILE *USER-PFA *BY-SYSTEM	
Name of the cache area	var(*LIST).CACHE-ID	S	<c-string 1..32>	
Cache medium of the cache area *MAIN-MEM=main memory cache area *GS=global storage cache area	var(*LIST).CACHE-MED	S	*MAIN-MEM *GS	
Segment size of the cache area (not for AutoDAB)	var(*LIST).CACHE-SEGMENT-SIZE	I	*UNDEF <integer 4..32>	
Requested size of the cache area	var(*LIST).CACHE-SIZE	I	<integer 1..8388608>	
Size of the cache area in: *KB=kilobytes *MB=megabytes	var(*LIST).SIZE-DIM	S	*KB *MB	
Current size of the cache area	var(*LIST).CACHE-SIZE-ACTUAL	I	<integer 1..8388608>	a
Size of the current cache area in: *KB=kilobytes *MB=megabytes	var(*LIST).SIZE-DIM-ACTUAL	S	*KB *MB	a
Location of the cache area in main memory	var(*LIST).MEMORY	S	*ABOVE *BELOW *ANY	a
Share of the cache occupied by the data areas	var(*LIST).CACHE-USE-RATIO	I	<integer 0..100>	3c
Share of the cache occupied by the files for AREA=*BY-SYSTEM	var(*LIST).CACHED.CACHE-USE-RATIO	I	<integer 0..100>	3d
Number of cached files for AREA=*BY-SYSTEM	var(*LIST).CACHED.NUM-OF-FILE	i	<integer 0..4294967296>	3d
Number of read accesses to the cached files for AREA=*BY-SYSTEM	var(*LIST).CACHED.NUM-OF-READ	I	<integer 0..4294967296>	3d
Number of write accesses to the cached files for AREA=*BY-SYSTEM	var(*LIST).CACHED.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Read hit rate of the cached files for AREA=*BY-SYSTEM	var(*LIST).CACHED.READ-HIT-RATIO	I	<integer 0..100>	3d

Output information	Name of the S variable	T	Contents	Condition
Write hit rate of the cached files for AREA=*BY-SYSTEM	var(*LIST).CACHED.WRITE-HIT-RATIO	I	<integer 0..100>	3d
Caching mode *READ=read caching *READ-WRITE=read/write caching *WRITE=write caching *USER-PFA=depending on file attributes *BY-MED=depending on cache medium	var(*LIST).CACHING-MODE	S	*READ *READ-WRITE *WRITE *USER-PFA *BY-MED	
Number of closed files	var(*LIST).CLOSE.NUM-OF-FILES	I	<integer 0..4294967296>	3d
Number of read accesses to already closed files	var(*LIST).CLOSE.NUM-OF-READ	I	<integer 0..4294967296>	3d
Number of write accesses to already closed files	var(*LIST).CLOSE.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Read hit rate of closed files	var(*LIST).CLOSE.READ-HIT-RATIO	I	<integer 0..100>	3d
Write hit rate of closed files	var(*LIST).CLOSE.WRITE-HIT-RATIO	I	<integer 0..100>	3d
Share of cache occupied by file	var(*LIST).FILE(*LIST).CACHE-USE-RATIO	I	<integer 0..100>	2d
File name	var(*LIST).FILE(*LIST).F-NAME	S	<c-string 1..54>	1b
Size of file	var(*LIST).FILE(*LIST).F-SIZE	I	<integer 0..16777216>	1b
Number of read accesses	var(*LIST).FILE(*LIST).NUM-OF-READS	I	<integer 0..4294967296>	1b
Number of write accesses	var(*LIST).FILE(*LIST).NUM-OF-WRITES	I	<integer 0..4294967296>	1b
Read hit rate	var(*LIST).FILE(*LIST).READ-HIT-RATIO	I	<integer 0..100>	1b

Output information	Name of the S variable	T	Contents	Condition
<p>Status of the served file</p> <p>*Y=file or partial qualification currently served by DAB</p> <p>*Y(SHARED)=file on shared shared disk served</p> <p>*Y(STOP)=cache area just released</p> <p>*Y(PARTIAL)=file partly served, as it is distributed over multiple disks, not all of which are exclusively allocated</p> <p>*Y(MANDATORY)= file or partial qualification whose caching is explicitly specified with /MODIFY-DAB-CACHING.</p> <p>*Y(PINNED-DATA)=</p> <p>*N(PINNED-DATA)= cache data of the data area could not be saved in the last data area save run because of a hardware or software error</p> <p>*N(INVALID)=data description of DAB and of the disk space allocator no longer match because of errors</p> <p>*N(SHARED)=file on shared disk not served</p> <p>*N(CLOSE)=file is closed</p> <p>*N(DELETED)=file being deleted</p> <p>*N(SUSPENDED)=file excluded from caching because of poor cache utilization</p> <p>*N(CLOSE)=file is closed</p> <p>*N(DELETED)=file being deleted</p> <p>*N(SUSPENDED)=file excluded from caching because of poor cache utilization</p> <p>*N(REMOVED)=file or partial qualification explicitly excluded from caching with /MODIFY-DAB-CACHING</p> <p>*N=areas not being served by DAB at present</p> <p>*UNDEF=status could not be determined</p> <p>...</p>	<p>var(*LIST).FILE(*LIST).SERVICE</p>	<p>S</p>	<p>*Y</p> <p>*Y(SHARED)</p> <p>*Y(STOP)</p> <p>*Y(PARTIAL)</p> <p>*Y(MANDATORY)</p> <p>*Y(PINNED-DATA)</p> <p>*N(PINNED-DATA)</p> <p>*N(INVALID)</p> <p>*N(SHARED)</p> <p>*N(CLOSE)</p> <p>*N(DELETED)</p> <p>*N(SUSPENDED)</p> <p>*N(REMOVED)</p> <p>*N</p> <p>*UNDEF</p> <p>*N(ENCRYPTED)</p> <p>*Y(ENCRYPTED)</p> <p>*N(CLOSED/CRYPT)</p> <p>*Y(SHARED/CRYPT)</p> <p>*Y(MAND/CRYPT)</p> <p>*Y(SH/MAND/CRYPT)</p> <p>)</p>	<p>1b</p>

Output information	Name of the S variable	T	Contents	Condition
(Continued) *N(ENCRYPTED)= *Y(ENCRYPTED)=no write caching is performed for the encrypted file. *N(CLOSED/CRYPT)=encrypted file is closed. No write caching. *Y(SHARED/CRYPT)=encrypted file on shared disk is being served but is not write-cached. *Y(MAND/CRYPT)=encrypted file is cached explicitly but is not write-cached. *Y(SH/MAND//CRYPT)=encrypted file on shared disk is cached explicitly but is not write-cached.	var(*LIST).FILE(*LIST).SERVICE	S	*Y *Y(SHARED) *Y(STOP) *Y(PARTIAL) *Y(MANDATORY) *Y(PINNED-DATA) *N(PINNED-DATA) *N(INVALID) *N(SHARED) *N(CLOSE) *N(DELETED) *N(SUSPENDED) *N(REMOVED) *N *UNDEF *N(ENCRYPTED) *Y(ENCRYPTED) *N(CLOSED/CRYPT) *Y(SHARED/CRYPT) *Y(MAND/CRYPT) *Y(SH/MAND/CRYPT))	1b
Write hit rate	var(*LIST).FILE(*LIST).WRITE-HIT-RATIO	I	<integer 0..100>	1b
Cache utilization level as of which the write data of the cache area is saved to disk NO=the data is not saved under threshold control. Data is only saved when the cache area is released or when the disk is no longer allocated AT-LOW-FILL=threshold-controlled saving when 25% of the cache area is filled with write data AT-HIGH-FILL=threshold-controlled saving when 75% of the cache area is filled with write data	var(*LIST).FORCE-OUT	S	*UNDEF *NO *AT-LOW-FILL *AT-HIGH-FILL	
GS unit specification 1=mono partition on GS unit 1 2=mono partition on GS unit 2 DUAL=dual partition	var(*LIST).GS-UNIT	S	1 2 *DUAL	e
Number of served files	var(*LIST).NUM-OF-FILE	I	<integer 0..4294967296>	3b
Number of read accesses	var(*LIST).NUM-OF-READ	I	<integer 0..4294967296>	3c

Output information	Name of the S variable	T	Contents	Condition
Number of write accesses	var(*LIST).NUM-OF-WRITE	I	<integer 0..4294967296>	3c
Free space in the GS partition	var(*LIST).PART(*LIST).FREE-SIZE	I	<integer 1..32767>	e
GS partition ID	var(*LIST).PART(*LIST).PART-ID	S	<c-string 1..8>	e
Status of the GS partition *OPERABLE=partition is available *MONO=a GS unit error was detected during access to a dual partition; cache data is still readable, but new data imports are suppressed *DEFECT=an error was detected during access to the cache medium, cache data may no longer be readable	var(*LIST).PART(*LIST).STA	S	*OPERABLE *MONO *DEFECT	e
Status of cached private disks	var(*LIST).PRIV(*LIST).SERVICE	S	*N *Y *Y(SHARED) *N(SHARED) *UNDEF	1d
VSN of cached private disks for AREA=*BY-SYSTEM	var(*LIST).PRIV(*LIST).VOL	S	<c-string 1..6>	1d
Pubset ID of the cached pubsets for AREA=*BY-SYSTEM	var(*LIST).PUBSET(*LIST).PUBSET	S	<cat-id>	1d
Type of pubset	var(*LIST).PUBSET(*LIST).PUBSET-TYPE	S	*SINGLE-FEATURE *SYS-MANAGE *VOLSET *UNDEF	1d
Status of pubset	var(*LIST).PUBSET(*LIST).SERVICE	S	*N *Y *Y(SHARED) *N(SHARED) *UNDEF *UNDEF	1d
Share of cache allocated (random access)	var(*LIST).RANDOM.CACHE-USE-RATIO	I	<integer 0..100>	3d
Number of files recognized as having random processing for AREA=*BY-SYSTEM	var(*LIST).RANDOM.NUM-OF-FILE	I	<integer 0..4294967296>	3d
Number of read accesses (random access)	var(*LIST).RANDOM.NUM-OF-READ	I	<integer 0..4294967296>	3d
Number of write accesses (random access)	var(*LIST).RANDOM.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Read hit rate (random access)	var(*LIST).RANDOM.READ-HIT-RATIO	I	<integer 0..100>	3d
Write hit rate (random access)	var(*LIST).RANDOM.WRITE-HIT-RATIO	I	<integer 0..100>	3d

Output information	Name of the S variable	T	Contents	Condition
Read hit rate	var(*LIST).READ-HIT-RATIO	I	<integer 0..100>	3c
Temporary caching mode following failure of a GS unit in cache areas with MEDIUM= *GS(GS-UNIT=*DUAL) and READ error handling	var(*LIST).READ-ONLY	S	*Y	f
Share of cache allocated (sequential access)	var(*LIST).SEQ.CACHE-USE-RATIO	I	<integer 0..100>	3d
Number of files recognized as having sequential processing for AREA=*BY-SYSTEM	var(*LIST).SEQ.NUM-OF-FILE	I	<integer 0..4294967296>	3d
Number of write accesses (sequential access)	var(*LIST).SEQ.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Number of read accesses (sequential access)	var(*LIST).SEQ.NUM-OF-READ	I	<integer 0..4294967296>	3d
Read hit rate (sequential access)	var(*LIST).SEQ.READ-HIT-RATIO	I	<integer 0..100>	3d
Write hit rate (sequential access)	var(*LIST).SEQ.WRITE-HIT-RATIO	I	<integer 0..100>	3d
Defines whether shared allocated disks are to be served for read caching (only for AREA=*VOLUME / *FILE) *TRUE=information is output *FALSE=information is not output	var(*LIST).SHARED-DISK-SUP	B	*UNDEF *TRUE *FALSE	
Displays whether a cache area was set up with CACHE-SIZE=*BY-FILE; only for cache areas set up with AREA=*FILE, otherwise displayed under *UNDEF	var(*LIST).SIZE-BY-FILE	B	*UNDEF *TRUE *FALSE	
Number of uncached files	var(*LIST).UNCACHED.NUM-OF-FILE	I	<integer 0..4294967296>	3d
Read accesses to uncached files	var(*LIST).UNCACHED.NUM-OF-READ	I	<integer 0..4294967296>	3d
Write accesses to uncached files	var(*LIST).UNCACHED.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Share of cache allocated (access behavior unknown)	var(*LIST).UNDEF.CACHE-USE-RATIO	I	<integer 0..100>	3d
Number of files for AREA=*BY-SYSTEM whose access behavior could not yet be determined	var(*LIST).UNDEF.NUM-OF-FILE	I	<integer 0..4294967296>	3d
Number of read accesses (access behavior unknown)	var(*LIST).UNDEF.NUM-OF-READ	I	<integer 0..4294967296>	3d

Output information	Name of the S variable	T	Contents	Condition
Number of write accesses (access behavior unknown)	var(*LIST).UNDEF.NUM-OF-WRITE	I	<integer 0..4294967296>	3d
Write hit rate	var(*LIST).WRITE-HIT-RATIO	I	<integer 0..100>	3c

Examples*Example 1 (output of various cache areas)*

```

/show-dab-caching cache-id=gsaudrw,
information=*system-cached-files(caching=*any)

CACHE-ID      = GSAUDRW
CACHE-MEDIUM = GS (GS-UNIT=DUAL)
PARTITION-ID  = DABDB302(STATE = OPERABLE)          FREE-SIZE = 1484 MB
CACHE-SIZE    = 400 MB
CACHING-MODE  = READ-WRITE(FORCE-OUT=AT-LOW-FILLING)
-----
CURRENT STATISTIC INTERVAL SET UP AT:   <date> <time>
-----
CATID/VSN     TYPE           SERVICE
=====
:X:           SF-PUBSET       Y
:B304:       SF-PUBSET       Y
WORK02       PRIVATE-DISK      Y
WORK01       PRIVATE-DISK      Y
-----
FILE  AREA      CACHE-USE  SERVICE  # OF READS  RD-HIT # OF WRITES  WR-HIT
=====
:A302:$TSOS.ARCHIVE.SAVE.FILE.010804.203831.WORK02
      1-1404      0%      N(CLOSED)
:B304:$DB124NK.E.Q1UTMGE1.UDOMI
      1-3         0%      Y           160    100%         2    100%
:B304:$DB124NK.FBUTMGE1.KDCA
      1-22314     11%     Y           314    98%        3124966  100%
:B304:$DB124NK.I.XSUTM.UDS.UTM.CTRL
      1-39        0%      Y           47     98%         33    100%
:B304:$DB124NK.L.Q1UTMGE1.UDOMI.9J7Y
      1-2298      1%      Y           0      2147    100%
:B304:$DB124NK.MA.LEIA.BASE
      1-580224    0%      N(SUSPENDED)
...
:B304:$DB124NK.SDF.USER.SYNTAX
      1-279      0%      Y           371    94%         0
:B304:$DB124NK.UDS.ENTER.9J6W.ST001
      1-3         0%      Y           3     100%         1    100%
:B304:$DB125SQL.SYSMSG.A.ESQL.011
      1-30        0%      Y           159    95%         0
:B304:$TSOS.TSOSCAT
      1-10002     2%      Y          375435  98%         543   100%
...
:X:$TSOS.TSOSCAT
      1-1002     0%      Y           65     97%         0

```

/show-dab-caching cache-id=gsaudrw,inf=*summary

CACHE-ID = GSAUDRW
 CACHE-MEDIUM = GS (GS-UNIT=DUAL)
 PARTITION-ID = DABDB302(STATE = OPERABLE) FREE-SIZE = 1484 MB
 CACHE-SIZE = 400 MB
 CACHING-MODE = READ-WRITE(FORCE-OUT=AT-LOW-FILLING)

 CURRENT STATISTIC INTERVAL SET UP AT: <date> <time>

	# OF FILES	CACHE-USE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
CACHED	138	87%	540696	84%	3723420	100%
(SEQUENTIAL)	98	17%	123037	98%	86534	100%
(RANDOM)	28	67%	417112	80%	3443284	100%
(UNDEFINED)	12	3%	547		20	
CLOSED			12453696	89%	4534536	100%
UNCACHED	1		24534		345	

/show-dab-caching inf=*all

CACHE-ID = BUFFER#001
 CACHE-MEDIUM = M-M
 CACHE-SIZE = 100 MB REDUCED TO 66 MB (ABOVE)
 CACHING-MODE = BY-MEDIUM (FORCE-OUT=AT-LOW-FILLING)

 CURRENT STATISTIC INTERVAL SET UP AT: <date> <time>

CATID/VSN	TYPE	SERVICE
=====	=====	=====
:RATS:	SF-PUBSET	Y
:CAM4:	SF-PUBSET	Y
WORK01	PRIVATE-DISK	Y

FILE	AREA	CACHE-USE	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
=====	=====	=====	=====	=====	=====	=====	=====
:CAM4:\$RZV120.PROC.UPDATE							
1-1224		1%	Y	208	54%	0	
:CAM4:\$RZV120.S.116.06V7.UL							
1-72		0%	Y	0		59	100%
:CAM4:\$TSOS.TSOSCAT							
1-8193		0%	Y	0		0	
:RATS:\$ATSSRPM1.SRPM.GRUPPEN.LIB.V11							
1-10881		1%	Y	1179	84%	0	
:RATS:\$ATSSRPM1.SRPM.START.PRC.V11							
1-3		0%	Y	6	83%	0	
:RATS:\$ATSSRPM1.SRPM.TOOLS.LIB.V11							
1-843		0%	Y	15949	100%	0	
:RATS:\$TSOS.SYSCAT.GUARDS							
1-48		0%	Y	2	50%	0	

```
:RATS:$TSOS.TSOSCAT
      1-8193      0%      Y      5      0%      1908      0%
```

```
/show-dab-caching cache-id=buffer6
```

```
CACHE-ID      = BUFFER6
CACHE-MEDIUM = M-M
CACHE-SIZE    = 16416 KB (BY-FILE)      SEGMENT-SIZE = 32 KB
CACHING-MODE = READ
```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:  <date> <time>
-----
```

FILE	AREA	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
====	====	=====	=====	=====	=====	=====
:RQH1:\$TSOS.TSOSCAT						
	1-8196	Y	316	84%	137	0%

Example 2 (output in S variables)

The following example illustrates how the information for a SHOW-DAB-CACHING command is saved in a previously declared S variable (in this case LVAR3).

```
/declare-variable lvar3(type=*structure),multiple=*list
/exec-cmd (show-dab-caching cache-id=*all),struc-out=lvar3
```

```
CACHE-ID      = CV62
CACHE-MEDIUM = GS (GS-UNIT=DUAL)
PARTITION-ID = DABDS062(STATE = OPERABLE)      FREE-SIZE = 0 MB
CACHE-SIZE    = 300 MB
CACHING-MODE = USER-PFA (FORCE-OUT=NO)        DESTAGED = 95 %
```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:  <date> <time>
-----
```

FILE	AREA	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
====	====	=====	=====	=====	=====	=====
:S062:\$CMB10.DO.MEMREK						
	1-6	Y(SHARED)	53	96%	0	
:S062:\$CMB10.WORKCAT.ORNELVM2.SRC						
	1-2718	Y(SHARED)	14	100%	1358	100%
:S062:\$QM124NK1.DBV40WN1.DBSTAT						
	1-24	Y(SHARED)	408	100%	372	100%
:S062:\$QM124NK1.DBV40WN1.DBSTAT.SAVE						
	1-24	Y(SHARED)	120	100%	132	100%
:S062:\$QM124NK1.DBV40WN1.RLOG.010429113506.1						
	1-192	Y(SHARED)	0		96	100%
:S062:\$QM124NK1.DBV40WN1.RLOG.010429113506.2						
	1-192	Y(SHARED)	0		96	100%
:S062:\$QM124NK1.P.CRE.GSVOL						
	1-900	Y(SHARED)	118	90%	30	100%

```

:S062:$QM124NK1.S.LST.0ANW.2008-04-29.110323
      1-1560      Y(SHARED)      0      1538      100%
:S062:$QM124NK1.S.LST.0ANX.2008-04-29.110323
      1-1560      Y(SHARED)      0      1538      100%
:S062:$QM124NK1.S.LST.0APN.2008-04-29.110323
      1-1560      Y(SHARED)      0      1538      100%
:S062:$QM124NK1.S.LST.0APP.2008-04-29.110323
      1-1560      Y(SHARED)      0      1538      100%
    
```

```

CACHE-ID      = 1
CACHE-MEDIUM = M-M
CACHE-SIZE    = 40 MB
CACHING-MODE = USER-PFA (FORCE-OUT=NO)      DESTAGED = 97 %
    
```

CURRENT STATISTIC INTERVAL SET UP AT: <date> <time>

FILE	AREA	SERVICE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
====	====	=====	=====	=====	=====	=====
:S062:\$CMB10.PAM10.TEST-FILE.PRF41	1-120	Y(SHARED)	791816	100%	791876	100%
:S062:\$CMB10.PAM10.TEST-FILE.PRF42	1-120	Y(SHARED)	674320	100%	674380	100%
:S062:\$CMB10.PAM10.TEST-FILE.PRF43	1-120	Y(SHARED)	707604	100%	707664	100%
:S062:\$CMB10.PAM10.TEST-FILE.PRF44	1-120	Y(SHARED)	719550	100%	719610	100%

/show-var lvar3

```

LVAR3(*LIST).CACHE-ID = CV62
LVAR3(*LIST).CACHE-MED = *GS
LVAR3(*LIST).GS-UNIT = *DUAL
LVAR3(*LIST).PART(*LIST).PART-ID = DABDS062
LVAR3(*LIST).PART(*LIST).STA = *OPERABLE
LVAR3(*LIST).PART(*LIST).FREE-SIZE = 0
LVAR3(*LIST).CACHE-SIZE = 300
LVAR3(*LIST).SIZE-DIM = *MB
LVAR3(*LIST).SIZE-BY-FILE = *UNDEF
LVAR3(*LIST).CACHE-SEGMENT-SIZE = *UNDEF
LVAR3(*LIST).CACHING-MODE = *USER-PFA
LVAR3(*LIST).FORCE-OUT = *NO
LVAR3(*LIST).SHARED-DISK-SUP = *UNDEF
LVAR3(*LIST).STATIS(*LIST).STA = *ACTIVE
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-DATE = 2008-12-21
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-TIME = 00:00:00
LVAR3(*LIST).AREA-TYPE = *BY-SYSTEM
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.DO.MEMREK
LVAR3(*LIST).FILE(*LIST).F-SIZE = 6
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
    
```

```
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 53
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 96
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 0
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 96
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.WORKCAT.ORNELVM2.SRC
LVAR3(*LIST).FILE(*LIST).F-SIZE = 2718
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 14
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 1358
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.DBV40WN1.DBSTAT
LVAR3(*LIST).FILE(*LIST).F-SIZE = 24
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 408
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 372
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.DBV40WN1.DBSTAT.SAVE
LVAR3(*LIST).FILE(*LIST).F-SIZE = 24
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 120
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 132
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.DBV40WN1.RLOG.
010429113506.1
LVAR3(*LIST).FILE(*LIST).F-SIZE = 192
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 96
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.DBV40WN1.RLOG.
010429113506.2
LVAR3(*LIST).FILE(*LIST).F-SIZE = 192
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 96
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.P.CRE.GSVOL
LVAR3(*LIST).FILE(*LIST).F-SIZE = 900
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 118
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 90
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 30
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 90
```



```
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.S.LST.0ANW.2008-04-29.110323
LVAR3(*LIST).FILE(*LIST).F-SIZE = 1560
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 1538
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.S.LST.0ANX.2008-04-29.110323
LVAR3(*LIST).FILE(*LIST).F-SIZE = 1560
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 1538
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.S.LST.0APN.2008-04-29.110323
LVAR3(*LIST).FILE(*LIST).F-SIZE = 1560
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 1538
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$QM124NK1.S.LST.0APP.2008-04-29.110323
LVAR3(*LIST).FILE(*LIST).F-SIZE = 1560
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 0
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 1538
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 0
LVAR3(*LIST).CACHE-ID = 1
LVAR3(*LIST).CACHE-MED = *MAIN-MEM
LVAR3(*LIST).CACHE-SIZE = 40
LVAR3(*LIST).SIZE-DIM = *MB
LVAR3(*LIST).SIZE-BY-FILE = *UNDEF
LVAR3(*LIST).CACHE-SEGMENT-SIZE = *UNDEF
LVAR3(*LIST).CACHING-MODE = *USER-PFA
LVAR3(*LIST).FORCE-OUT = *NO
LVAR3(*LIST).SHARED-DISK-SUP = *UNDEF
LVAR3(*LIST).STATIS(*LIST).STA = *ACTIVE
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-DATE = 2008-12-21
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-TIME = 00:00:00
LVAR3(*LIST).AREA-TYPE = *BY-SYSTEM
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.PAMIO.TEST-FILE.PRF41
LVAR3(*LIST).FILE(*LIST).F-SIZE = 120
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
```

```
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 791816
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 791876
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.PAMIO.TEST-FILE.PRF42
LVAR3(*LIST).FILE(*LIST).F-SIZE = 120
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 674320
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 674380
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.PAMIO.TEST-FILE.PRF43
LVAR3(*LIST).FILE(*LIST).F-SIZE = 120
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 707604
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 707664
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).F-NAME = :S062:$CMB10.PAMIO.TEST-FILE.PRF44
LVAR3(*LIST).FILE(*LIST).F-SIZE = 120
LVAR3(*LIST).FILE(*LIST).SERVICE = *Y(SHARED)
LVAR3(*LIST).FILE(*LIST).NUM-OF-READ = 719550
LVAR3(*LIST).FILE(*LIST).READ-HIT-RATIO = 100
LVAR3(*LIST).FILE(*LIST).NUM-OF-WRITE = 719610
LVAR3(*LIST).FILE(*LIST).WRITE-HIT-RATIO = 100
```

START-DAB-CACHING

Create ADM-PFA DAB cache areas

Domain: SYSTEM-TUNING

Privileges: TSOS

Function

The `/START-DAB-CACHING` command is used to create DAB cache areas (also referred to as ADM-PFA cache areas). The buffered database can be selected automatically by AutoDAB (specification of `AREA=*BY-SYSTEM`) or manually (specification of `AREA=*FILE(...)`).

The exact DAB method of operation can be specified for each cache area separately as follows:

- selecting the data areas to be served
- selecting the cache area with regard to storage type (MM or GS) and size of the cache (`CACHE-SIZE`)
- defining the cache area ID (`CACHE-ID`)
- defining the caching mode (read, write or read/write cache)
- defining the size of cache segments (4, 8, 16 or 32 KB)
- defining the data backup level with or without defining threshold-controlled saving to disk
- implicitly defining the caching technique via the `CACHE-SIZE` operand (displacement according to LRU or resident buffering)
- defining the location of the cache area and its management data (resident below or non-resident above the minimum main memory size)
- defining whether data on shared private disks and shared pubsets is to be served as well.



The input length of the `/START-DAB-CACHING` command is restricted to a maximum of 4096 characters (including blanks and comments).

The changeable attributes of a configured DAB cache area can be modified dynamically using the `/MODIFY-DAB-CACHING` command (see [page 89](#)).

Format

START-DAB-CACHING	Alias: SRDABC
<p>AREA = *BY-SYSTEM(...) / *FILE(...)</p> <p>*BY-SYSTEM(...)</p> <ul style="list-style-type: none"> PUBSET = *NO / list-poss(100): <cat-id> ,PRIVATE-VOLUME = *NO / list-poss(100): <vsn> <p>*FILE(...)</p> <ul style="list-style-type: none"> FILE-AREA = *NO / list-poss(16): <filename 1..54 without-vers> <p>,CACHE-SIZE = <integer 1..8388608>(...) / *BY-FILE</p> <p><integer 1..8388608>(...)</p> <ul style="list-style-type: none"> DIMENSION = *KILOBYTE / *MEGABYTE <p>,CACHE-ID = *STD / <name 1..32></p> <p>,CACHING-MODE = *READ / *WRITE / *READ-WRITE / *BY-CACHE-MEDIUM</p> <p>,CACHE-MEDIUM = *MAIN-MEMORY (...) / *GS(...)</p> <p>*MAIN-MEMORY(...)</p> <ul style="list-style-type: none"> CACHE-SEGMENT-SIZE = *32 / *4 / *8 / *16 ,MEMORY = *STD / *ABOVE-MIN-MEM-SIZE / *ANY / *BELOW-MIN-MEM-SIZE <p>*GS(...)</p> <ul style="list-style-type: none"> CACHE-SEGMENT-SIZE = *32 / *4 / *8 / *16 ,GS-UNIT = 1 / <integer 1..2> / *DUAL <p>,FORCE-OUT = *AT-LOW-FILLING / *AT-HIGH-FILLING / *NO</p> <p>,SHARED-DISK-SUPPORT = *NO / *YES</p>	

Operands

AREA = *BY-SYSTEM / *FILE(...)

Assigns data areas either automatically or manually at file level.

AREA = *BY-SYSTEM(...)

Assignment is made based on pubset/private disks. AutoDAB selects the files to be served on these volumes.

PUBSET = *NO / list-poss(100): <cat-id>

Specifies the pubsets/volume sets to be served with automatic caching by DAB.

PUBSET = *NO

No pubsets are to be served. The pubsets must be added to the data area with /MODIFY-DAB-CACHING.

PUBSET = list-poss(100): <cat-id>

Specifies the pubsets/volume sets whose files are to be served by DAB. The catalog IDs are to be specified (up to 100).



Just one SM pubset may be specified for a data area with CACHE-MEDIUM=*GS(...) and CACHING-MODE=*WRITE, *READ-WRITE or *BY-CACHE-MEDIUM. No further data areas (SF pubsets, additional volume sets or private disks) may be specified.

PRIVATE-VOLUME = *NO / list-poss(100): <vsn>

Specifies the private disks to be served with automatic caching by DAB.

PRIVATE-VOLUME = *NO

No private disks are to be served. Specification of PUBSET is then mandatory.

PRIVATE-VOLUME = list-poss(100): <vsn>

Specifies the private disks whose files are to be served by AutoDAB. The VSNs of the private disks (up to 100) containing the files are to be specified.

AREA = *FILE(...)

Assigns data areas at file level.

FILE-AREA = *NO

No file specified.

The files must be added to the data area with /MODIFY-DAB-CACHING.

FILE-AREA = list-poss(16): <filename 1..54 without-vers>

The files specified here are to be served by DAB immediately. Up to 16 files may be specified. Further files can, however, be added to the data area using

/MODIFY-DAB-CACHING.

CACHE-SIZE =

Defines the size of the cache area.



The more memory used in the main memory cache medium for caching, the less memory there is available for paging activities. A rise in the paging rate reduces the performance gain for DAB and should therefore be avoided.

CACHE-SIZE = <integer 1..8388608>(…)

Size of the cache area which is to be used for buffering the data areas identified by the AREA operand.

DIMENSION = *KILOBYTE / *MEGABYTE

Determines whether the size of the cache area is specified in KB or MB.



- The more memory used in the main memory cache medium for caching, the less memory there is available for paging activities. A rise in the paging rate reduces the performance gain for DAB and should therefore be avoided.
- The value should be a multiple of 32 KB. If it is not, DAB rounds it down to the next lowest multiple of 32. The allocation for key and management data is added to this value.
- The value of <integer> must be smaller than 7/8 of the main memory which is still available for paging or the storage space still available in the GS partition, or the installation-specific value must be adjusted.

CACHE-SIZE = *BY-FILE

The size of the cache area depends on the data areas specified with the FILE-AREA operand.



- CACHE-SIZE=*BY-FILE is only permitted if the data areas to be served are specified via the FILE-AREA operand.
- Any expansion of the disk storage allocation for these files effected while they are being served by DAB automatically expands the associated cache area, provided there is cache space available (with GS subject to the partition size).

CACHE-ID = *STD / <name 1..32>

Defines an identifier for the cache area.

CACHE-ID = *STD

Default value: BUFFER#iii

(iii = lowest internal DAB number not yet assigned.)

CACHE-ID = <name 1..32>

Identifier assigned to the new cache area. <name> must not contain any special character other than '#', '@' and '\$' and must start with a letter.

CACHING-MODE = *READ / *WRITE / *READ-WRITE / *BY-CACHE-MEDIUM

Defines the caching mode to be used.



CACHING-MODE=*WRITE or CACHING-MODE=*READ-WRITE is permissible if the data areas to be served are specified via the FILE-AREA operand.

CACHING-MODE = *READ

Caching mode is read.

CACHING-MODE = *WRITE

Caching mode is write.

CACHING-MODE = *READ-WRITE

Caching mode is read/write.

CACHING-MODE = *BY-CACHE-MEDIUM

The setting for read or read/write caching is dependent on the cache medium and the data area specification.

- Manual caching at file level (specification AREA=*FILE):
The READ caching mode is set for main memory (MAIN-MEMORY), and the READ-WRITE caching mode for global storage (GS).
- Automatic caching (specification AREA=*BY-SYSTEM):
The caching mode set depends on the cache medium and whether the file is temporary or permanent.
For main memory (MAIN-MEMORY), the READ caching mode is set for permanent files and READ-WRITE for temporary files. Temporary files are saved to disk when they are closed to ensure failsafe caching.
READ-WRITE is set for permanent and temporary files for global storage (GS).

CACHE-MEDIUM = *MAIN-MEMORY(...) / *GS(...)

Defines the cache medium to be used.

CACHE-MEDIUM = *MAIN-MEMORY(...)

The cache area is to be set up in main memory.

CACHE-SEGMENT-SIZE = *32 / *4 / *8 / *16

Defines the size of segments in the new cache area in KB. The operand is ignored for cache areas with automatic caching (AREA=*BY-SYSTEM).

MEMORY = *STD / *ANY / *BELOW-MIN-MEM-SIZE / *ABOVE-MIN-MEM-SIZE

Defines the system in which dynamic main memory reconfiguration is possible, and the location of the cache area and its management data.



On systems on which dynamic main memory reconfiguration is possible it can occur that cache areas with MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE do not achieve the size specified in the CACHE-SIZE operand, see the notes on [page 135](#).

On systems on which **no** main memory reconfiguration is possible, the operand values *ANY / *BELOW-MIN-MEM-SIZE / *ABOVE-MIN-MEM-SIZE are equivalent to each other; but sizes may be changed in the event of memory saturation. No size adjustment takes place with *STD.

MEMORY = *STD

The cache area and its management data are (as in earlier DAB versions) created as resident below the minimum main memory size.

The size of the cache area must comply with the following formula when it is created:

$$\text{CACHE-SIZE} \leq 7/8 * (P - 5) \text{ MB}$$

where P specifies (in MB):

- the size of the pageable main memory in the system when no main memory reconfiguration is possible
- the size of the pageable main memory below the minimum main memory size when main memory reconfiguration is possible

MEMORY = *ANY

The cache area and its management data are (preferably) created above and below the minimum main memory size.

When the main memory is reconfigured or the minimum main memory size is increased, the size of the cache areas is also adjusted. If necessary, the size of the cache areas is reduced to 0. The main memory size of a system can thus be handled flexibly.

The size of the cache area must comply with the following formula when it is created:

$$\text{CACHE-SIZE} \leq 7/8 * (P - 64) \text{ MB}$$

where P specifies (in MB) the size of the pageable main memory in the system.

MEMORY = *ABOVE-MIN-MEM-SIZE

The cache area and its management data are created only in the size which is possible above the minimum main memory size.

When the main memory is reconfigured or the minimum main memory size is increased, the size of the cache areas is also adjusted.

The size of the cache area must comply with the following formula when it is created:

$$\text{CACHE-SIZE} \leq 7/8 * (P - 64) \text{ MB}$$

where P specifies (in MB) the size of the pageable main memory above the minimum main memory size.

If there is no pageable main memory above the minimum main memory size, the cache area is created with the size 0. When the main memory is then enlarged later, it will be expanded to the size specified in the CACHE-SIZE operand.

MEMORY = *BELOW-MIN-MEM-SIZE

The cache area and its management data are created as resident below the minimum main memory size.

The size of the cache area must comply with the following formula:

$$\text{CACHE-SIZE} \leq 7/8 * (P - 64) \text{ MB}$$

where P specifies (in MB):

- the size of the pageable main memory in the system when no main memory reconfiguration is possible
- the size of the pageable main memory below the minimum main memory size when main memory reconfiguration is possible

When a higher value is specified for the cache size in the CACHE-SIZE operand, a cache size in accordance with the formula above is selected. If the minimum main memory size is increased later, the size of the cache area is increased to the desired cache size in accordance with the formula above.

In the case of main memory saturation, the cache size is automatically reduced. When the main memory saturation has been resolved, the cache size is increased again.

CACHE-MEDIUM = *GS(...)

The cache area is to be set up in global storage.

CACHE-SEGMENT-SIZE = *32 / *4 / *8 / *16

Defines the size of segments in the new cache area in KB. The operand is ignored for cache areas with automatic caching (AREA=*BY-SYSTEM).

GS-UNIT = 1 / <integer 1..2> / *DUAL

Specifies the logical address of the GS unit where the cache area is to be set up (currently restricted to the values 1 and 2).

Specifying “*DUAL” causes the cache area to be operated with dual recording, i.e. user and management data of the cache area is simultaneously kept in two different GS units.

FORCE-OUT = *AT-LOW-FILLING / *AT-HIGH-FILLING / *NO

Specifies whether save runs are to be triggered by a threshold value. This operand is relevant only with caching modes WRITE, READ-WRITE or BY-CACHE-MEDIUM (see “Notes”, point 4 on page 137). The operand is irrelevant for cache areas with automatic caching (AREA=*BY-SYSTEM) (see “Automatic FORCE-OUT correction” on page 35).

FORCE-OUT = *AT-LOW-FILLING

Threshold-controlled save runs are to be performed to keep the number of cache segments not saved to disk at any one time as small as possible. This specification triggers saving whenever 25% of the cache is filled with write data not yet saved to disk.

FORCE-OUT = *AT-HIGH-FILLING

Save runs for this cache area are to be restricted to the number required to ensure there are always enough segments available for caching. This specification triggers saving whenever 75% of the cache is filled with write data not yet saved to disk.

FORCE-OUT = *NO

No threshold-controlled save runs are to be performed to transfer data from this cache area to disk. Data transfer to disk is not performed until the cache area is released by means of the /STOP-DAB-CACHING command.

SHARED-DISK-SUPPORT = *NO / *YES

Determines whether data areas on disks used as shared private disks or shared pubsets are to be supported as well with AREA=*FILE/*VOLUME (see Notes 6 on page 137 and 7 on page 138). This applies to read mode only.

SHARED-DISK-SUPPORT = *NO

No data area is to be served that is located on a disk operated as a shared private disk or shared pubset when DAB starts service for the first time.

SHARED-DISK-SUPPORT = *YES

A disk data area is to be served even if the disk is operated as a shared private disk or shared pubset the next time DAB starts service.

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	0	NDB0021	No error. Cache area created with reduced size.
	1	CMD0202	Syntax or semantic error in command
	32	CMD0221	Internal SDF error
	64	CMD0216	Required privilege missing
	64	NDB0005	No authorization to invoke the command
	64	NDB0010	Wrong syntax file version
	64	NDB0012	DMS error
	64	NDB0013	Multiple file specification
	64	NDB0016	Cache memory bottleneck
	64	NDB0017	Memory bottleneck (cache or management data), global storage failure or system failure
	64	NDB0018	Cache ID already defined
	64	NDB0025	Volume not allocated
	64	NDB0026	File not cataloged
	64	NDB0027	Tape file or migrated file
	64	NDB0028	File/volume already being served
	64	NDB0032	File on PFA pubset cannot be served with /START-DAB-CACHING
	64	NDB0034	File on key disk cannot be added to the cache area without keys
	64	NDB0036	CACHE-SIZE=*BY-FILE permissible only in conjunction with file specification
	64	NDB0038	GS partition not defined
	64	NDB0039	No GS partition in the specified GS unit
	64	NDB0053	Cache size 0 not permissible for cache areas in the GS
	64	NDB0055	Data on GS volumes is not served
	64	NDB0061	Data of the volume cannot be served in two different GS partitions
	64	NDB0065	/STOP-SUBSYSTEM is active
	64	NDB0066	No default cache ID available
	64	NDB0071	File cannot be cached because it is encrypted and the cache is a write cache
	64	NDB0072	Unknown user ID
	64	NDB0080	Automatic caching of data in the home pubset is not permissible in a write or read/write cache

(SC2)	SC1	Maincode	Meaning
	64	NDB0081	In parallel HIPLEX, caching of home pubset data using cache medium GS in conjunction with write or read/write cache is not permissible
	64	NDB0098	Pubset not imported locally
	64	NDB0099	Pubset from GS volume is not served
	64	NDB0100	Pubset already being served
	64	NDB0101	More than one SM pubset specified for (RD)WR caching
	64	NDB0102	Volume is not a private disk
	64	NDB0103	File catalog cannot be served due to error
	64	NDB0106	Neither pubset nor private disk specified for AREA=*BY-SYSTEM
	64	NDB0107	Pubset specified more than once
	64	NDB0110	SM pubset does not contain a cacheable volume set
	64	NDB0126	GS partition is reserved for PFA cache areas of shared pubsets.
	64	NDB0162	Processing of a pubset was aborted when a particular volume was accessed
	64	NDB0163	Processing of a pubset was aborted when a volume was accessed
	64	NDB0164	Processing of a volume was aborted when this volume was accessed
	64	NDB0165	Processing of a pubset was aborted when a particular volume was accessed. It is not known how processing was concluded
	64	NDB0166	Processing of a pubset was aborted when a volume was accessed. It is not known how processing was concluded
	64	NDB0167	Processing of a volume was aborted when this volume was accessed. It is not known how processing was concluded
	64	NDB0177	The configuration of a cache area cannot be modified during main memory reconfiguration
	65	CMD2241	Subsystem not available
	128	CMD2280	Saturation problem

Examples

1. Serving several pubsets and private disks in the GS cache medium with automatic file selection. The buffer area is to be created in a dual GS partition.

```
/START-DAB-CACHING CACHE-ID=GSAUDRW,CACHE-SIZE=400(*M), -
/
/          CACHE-MEDIUM=*GS(GS-UNIT=*DUAL), -
/          CACHING-MODE=*READ-WRITE, -
/          AREA=*BY-SYSTEM(PUBSET=(B304,X), -
/          PRIVATE-VOLUME=(WORK01,WORK02))
% NDB0020 /START-DAB-CACHING COMMAND ACCEPTED. THE DAB CACHE BUFFER
'GSAUDRW' WAS INSTALLED WITH 400 MB FOR THE FOLLOWING FILES/PUBSETS:
% F I L E / P U B S E T
% -----
% :B304: (SYSTEM-CONTROLLED)
% :X:   (SYSTEM-CONTROLLED)
% WORK01 (SYSTEM-CONTROLLED)
% WORK02 (SYSTEM-CONTROLLED)
```

2. Serving several pubsets in the MM cache medium with automatic file and caching mode selection.

```
/START-DAB-CACHING AREA=*BY-SYSTEM(PUBSET=(CAM4,RATS), -
/
/          CACHE-SIZE=100(*MEGABYTE), -
/          CACHE-MEDIUM=*MAIN-MEMORY(MEMORY=*ABOVE-MIN-MEM-SIZE), -
/          CACHING-MODE=*BY-CACHE-MEDIUM
% NDB0021 /START-DAB-CACHING COMMAND ACCEPTED. THE DAB CACHE BUFFER
'BUFFER#001' WAS INSTALLED WITH 66 MB (INSTEAD OF THE TARGET SIZE OF
100 MB) FOR THE FOLLOWING FILES/PUBSETS:
% F I L E / P U B S E T
% -----
% :CAM4: (SYSTEM-CONTROLLED)
% :RATS: (SYSTEM-CONTROLLED)
```

3. Serving the TSOSCAT file catalog of the RATS pubset with MM cache medium in read mode (it is only meaningful to serve the TSOSCAT of a pubset in read mode if the memory is volatile).

- with a fixed cache size of 1 Mbyte

```
/START-DAB-CACHING AREA=*FILE(FILE-AREA=:RATS:TSOSCAT), -
/
/          CACHE-SIZE=1(DIMENSION=*MEGABYTE), -
/          CACHE-MEDIUM=*MAIN-MEMORY,CACHING-MODE=*READ
% NDB0020 /START-DAB-CACHING COMMAND ACCEPTED. THE DAB CACHE BUFFER
'BUFFER#001' WAS INSTALLED WITH 1 MB FOR THE FOLLOWING FILES/PUBSETS:
% F I L E / P U B S E T
% -----
% :RATS:$TSOS.TSOSCAT
```

- with a variable cache size in “resident buffering” mode. The served data area is buffered completely in the cache.

```

/START-DAB-CACHING AREA=*FILE(FILE-AREA=:RATS:TSOSCAT), -
/
          CACHE-SIZE=*BY-FILE,CACHE-MEDIUM=*MAIN-MEMORY, -
/
          CACHING-MODE=*READ
% NDB0020 /START-DAB-CACHING COMMAND ACCEPTED. THE DAB CACHE BUFFER
'BUFFER#001' WAS INSTALLED WITH 16416 KB FOR THE FOLLOWING
FILES/PUBSETS:
% F I L E / P U B S E T
% -----
% :RATS:$TSOS.TSOSCAT

```

4. Sample application for CACHE-MEDIUM=*MAIN-MEMORY(MEMORY=*ABOVE-MIN-MEM-SIZE)

Preparing a standby system under VM2000 which, if necessary, is to take over the load of a productive system (before the productive system is started):

- Set the system’s minimum main memory size and current main memory size to the same minimum size to ensure that the requirement of the standby system and the requirement for resident memory are covered when the productive load is taken over.
- Import the pubsets which are to be buffered after the load has been taken over.
- Configure the cache areas with `/START-DAB-CACHING ... ,CACHE-MEDIUM= *MAIN-MEMORY(MEMORY=*ABOVE-MIN-MEM-SIZE)`. The cache areas will be assigned the current size 0.
- In `/START-DAB-CACHING` also specify the pubsets (AutoDAB) or files (FILE-AREA cache) which are to be buffered.
- Export the pubsets which are to be buffered after the load has been taken over.
- When the load has been taken over, increase the size of the main memory. The cache areas will be extended accordingly.

As an alternative to b), d) and e) you can add the pubsets and files with `/MODIFY-DAB-CACHING` after the load has been taken over. Preparation of the standby system is then independent of the productive system run.

Notes

1. Dynamic modification of a DAB cache area is only possible with the `/MODIFY-DAB-CACHING` command. Repeated `/START-DAB-CACHING` commands with the same `CACHE-ID` value are permissible only if a `/STOP-DAB-CACHING` command with the same `CACHE-ID` was executed in between.
2. A `/START-DAB-CACHING` command is not accepted for further processing unless all disks on which data areas to be served are allocated, are not GS volumes and do not belong to a pubset that is buffered with DAB using PFA.
3. Notes on the `CACHE-MEDIUM = *MAIN-MEMORY(MEMORY=...)` operand:
 - Cache areas with `MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE` only make sense in systems in which main memory reconfiguration is possible, e.g. under VM2000.

Creating cache areas above the minimum main memory size (`MEMORY=*ANY/*ABOVE-MIN-MEM-SIZE`) enables the minimum main memory size of the (VM2000) system to be kept low. This permits the main memory of a (standby) guest system under VM2000 to be reduced during ongoing operation when the load is low and increased when the load increases.

- In the case of cache areas with `MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE` it can occur that the cache area is not created using the size specified in the `CACHE-SIZE` operand, but is operated with a smaller size, the current one. The current size is always less than or equal to the specified size.

The reason is that at the time the cache area was created not enough space was available in the main memory or that the DAB cache was decreased in size owing to a memory reduction (in the case of `MEMORY=*ABOVE-MIN-MEM-SIZE/*ANY`) or memory saturation (`MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE`). DAB specifies the current size of the cache area. After every implicit change, the current size is logged on the console with the message NDB0052 for cache areas whose size has been reduced. As in the case of a memory reduction a cache area with `MEMORY= *ANY/*ABOVE-MIN-MEM-SIZE` must in all cases first be completely detached and reattached after the memory reduction, it is possible that the old cache size will be attained once more. This is logged on the console with the message NDB0176.

The current (reduced) size of the cache area can be changed using the `/MODIFY-DAB-CACHING` command, see [page 89](#).

- When DAB caches with MEMORY=*ANY are used, performance can occasionally be impaired if the location of this cache memory needs to be changed in the main memory. This can occur in the following cases:
 - The cache memory could not be created immediately in the preferred area in the main memory because pages could not be displaced to the auxiliary memory quickly enough or the main memory was occupied by other cache areas with MEMORY=*ANY/*ABOVE-MIN-MEM-SIZE.
 - The cache memory must be displaced owing to resident memory requirements.
- In the case of cache areas in the main memory with MEMORY=*ANY/*ABOVE-MIN-MEM-SIZE, before main memory reduction takes place or before the minimum main memory size is increased, the entire cache area is first reduced to 0.

After the main memory has been reduced or the minimum main memory size has been increased, the size of these cache areas is recalculated and reduced by the relevant percentage.

Example: A main memory reduction of 20% also leads to the size of the cache areas being reduced by 20%. The same effect occurs when the main memory area above the minimum main memory size is reduced by increasing the minimum main memory size by 20%.

When the main memory is expanded (again), the size of the cache areas is also increased proportionately up to the size specified in /START- or /MODIFY-DAB-CACHING.

- When the memory size is insufficient, cache areas in the main memory with MEMORY=*BELOW-MIN-MEM-SIZE are created with a size which is less than that specified. When the minimum main memory size is increased, such cache areas are increased proportionately up to the size specified in /START- or /MODIFY-DAB-CACHING.
- In the case of main memory saturation, DAB reduces the size of the caches areas with MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/ *ABOVE-MIN-MEM-SIZE proportionately. The size is increased correspondingly when memory saturation is reduced.

The new current size of the cache areas is logged on the console with the message NDB0052.



When cache areas with MEMORY=*STD or cache areas from earlier DAB versions are to be converted to cache areas with MEMORY=*ANY/*BELOW-MIN-MEM-SIZE/*ABOVE-MIN-MEM-SIZE, the system should be assigned a minimum main memory size which is adapted to the sizes of the cache areas.

4. FORCE-OUT operand

Whether FORCE-OUT=*AT-HIGH-FILLING or FORCE-OUT=*AT-LOW-FILLING is advisable depends on various factors. With *AT-HIGH-FILLING, the load on the I/O system is lower than with *AT-LOW-FILLING. On the other hand, the fact that 75% of the cache is filled with data may result in a cache overflow if a large amount of write data has to be handled at a time.

FORCE-OUT=*AT-LOW-FILLING, on the other hand, may result in superfluous save operations which increase the I/O system load.

With write caching, a check should be performed to determine whether cache behavior is satisfactory with FORCE-OUT=*AT-HIGH-FILLING. In the event of cache overflows, FORCE-OUT=*AT-LOW-FILLING should be used instead.

FORCE-OUT=*AT-LOW-FILLING should always be used with read/write caching.

FORCE-OUT=*NO is recommended for resident buffering.

With automatic caching, any incorrect setting of the FORCE-OUT operand is corrected (see also [section "Automatic caching \(AutoDAB\)" on page 34](#)).

5. When an SM pubset is to be buffered in a write or read/write cache, the SM pubset must already have been specified with /START-DAB-CACHING. It cannot be added retroactively with /MODIFY-DAB-CACHING. No further SF or SM pubset is permitted in this cache area.

6. Caching shared volumes or shared pubsets

Automatic caching

The SHARED-DISK-SUPPORT operand is irrelevant for automatically selected cache areas (AREA=*BY-SYSTEM). DAB takes over control for caching data on shared volumes in this case and ensures data consistency by means of the following measures:

- Data areas on these volumes are only served with read caching. Caching is only enabled if the caching mode of the cache area concerned explicitly specifies read caching.
- Read caching a file on such a volume is not accepted if the file is processed in a mode which allows it to be updated in parallel by another server.

Non-automatic caching

Using local system caches to serve data areas extending beyond a given system (e.g. files on shared private disks or shared pubsets) is generally problematic. Using non-automatic DAB for such data areas is not advisable unless access to the areas served by DAB is reliably restricted to read-only access for all sharers. The following should therefore be noted when specifying the SHARED-DISK-SUPPORT operand:

- Data areas located on a disk operated in shared disk allocation mode are not served if the associated cache area is used as read/write or write cache.
 - The SHARED-DISK-SUPPORT operand is evaluated during the /START-DAB-CACHING command processing. DAB detects and processes any subsequently modified allocation mode.
7. The decision as to whether file-specific data areas (AREA=*FILE) on shared volumes are served depends on the parameter specifications defined by the user for a cache area. This results in the following differences when caching such volumes:
- If a shared volume is served from different cache areas it is possible that its subareas are served by one cache area and not served by the others.
 - Since the allocation state of a volume can change frequently during serving by a cache area it is possible that a data area is alternately buffered or not buffered.

STOP-DAB-CACHING

Release existing ADM-PFA DAB cache areas

Domain: SYSTEM-TUNING

Privileges: TSOS

Function

The /STOP-DAB-CACHING command serves to release existing ADM-PFA DAB cache areas, thus releasing all storage areas allocated when the cache area was set up. Any cache data not yet updated in external storage is saved.

The released cache areas can subsequently be reassigned by new /START-DAB-CACHING commands.

PFA cache areas cannot be released using the /STOP-DAB-CACHING command.

Format

STOP-DAB-CACHING	Alias: SPDABC
CACHE-ID = *ALL / <name 1..32> ,CACHE-MEDIUM = *ALL / *MAIN-MEMORY / *GS(...) *GS(...) PARTITION-ID = *ALL / <name 1..8>	

Operands

CACHE-ID = *ALL / <name 1..32>

Specifies the cache areas which are to be released.

CACHE-ID = *ALL

All cache areas existing in the specified cache medium are to be released.



The DAB subsystem remains loaded even if all cache areas are successfully released and no more PFA cache area exists.

CACHE-ID = <name 1..32>

The specified cache area is to be released.

CACHE-MEDIUM = *ALL / *MAIN-MEMORY / *GS (...)

The cache area specified with the CACHE-ID operand will not be released unless it is found in the cache medium specified with this operand.

CACHE-MEDIUM = *ALL

The cache area specified with the CACHE-ID operand is to be released irrespective of the cache medium it is found in.

CACHE-MEDIUM = *MAIN-MEMORY

The cache area specified with the CACHE-ID operand will not be released unless it is found in main memory.

CACHE-MEDIUM = *GS (...)

The cache area specified with the CACHE-ID operand will not be released unless it is found in global storage.

PARTITION-ID = *ALL / <name 1..8>

Specifies the DAB partition in global storage which is to be searched for the cache area identified by the CACHE-ID operand.

PARTITION-ID = *ALL

Information about the cache area identified by the CACHE-ID operand is to be displayed irrespective of the DAB partition in global storage in which it is found.

PARTITION-ID = <name 1..8>

Restricts the search for the cache area identified by the CACHE-ID operand to a specific DAB partition in global storage.



After a reconstruction of GS cache areas from another system, there may be multiple ADM-PFA cache areas with the same cache ID. These are distinguished by the name of the GS partition in which they are embedded.

Specifying CACHE-MEDIUM=*GS(PARTITION-ID=...) permits the cache area to be addressed unequivocally.

Specifying PARTITION-ID = *ALL causes all cache areas with the specified cache ID in all GS partitions to be released.

Return codes

(SC2)	SC1	Maincode	Meaning
	0	CMD0001	No error
	32	CMD0221	Internal SDF error
	64	CMD0216	Required privilege missing
	64	NDB0005	No authorization to invoke the command
	64	NDB0010	Wrong syntax file version
	64	NDB0019	Specified cache area not found in specified medium
	64	NDB0023	Cache area could not be released
	64	NDB0024	No cache area exists
	64	NDB0029	Cache area could not be completely saved to disk
	64	NDB0030	Cache area is not released to permit subsequent reconstruction
	64	NDB0051	Cache area cannot be disconnected from a GS partition while the data areas supported by DMS are still occupied
	64	NDB0057	Cache area cannot be detached from GS partition as volumes are still used by DMS
	64	NDB0062	System error at disconnection
	64	NDB0067	Cache cannot be released due to system error
	64	NDB0068	Cache area not found
	64	NDB0069	No cache area in the specified medium is currently active
	64	NDB0070	command collides with /STOP-DAB-CACHING processed in parallel
	64	NDB0154	STOP-DAB-CACHING aborted because of pubset import
	64	NDB0178	Processing of the command has been interrupted for a cache area. It is not known how processing was concluded
	65	CMD2241	Subsystem not available

Examples

```
/show-dab-caching inf=*summary
```

```
CACHE-ID      = BUFFER4
CACHE-MEDIUM = M-M
CACHE-SIZE    = 40 MB
CACHING-MODE = READ
```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:   <date> <time>
-----
```

	# OF FILES	CACHE-USE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
CACHED	2	2%	20249	99%	0	0%
(SEQUENTIAL)	0	0%	0	0%	0	0%
(RANDOM)	1	2%	20249	99%	0	0%
(UNDEFINED)	1	0%	0	0%	0	0%
CLOSED		0%	44	93%	0	0%
UNCACHED	0	0%	0	0%	0	0%

```
CACHE-ID      = BUFFER#001
CACHE-MEDIUM = M-M
CACHE-SIZE    = 20 MB
CACHING-MODE = READ-WRITE(FORCE-OUT=AT-LOW-FILLING)
```

```
-----
CURRENT STATISTIC INTERVAL SET UP AT:   <date> <time>
-----
```

	# OF FILES	CACHE-USE	# OF READS	RD-HIT	# OF WRITES	WR-HIT
CACHED	50	42%	2206	86%	2185	44%
(SEQUENTIAL)	2	7%	455	95%	0	0%
(RANDOM)	16	34%	1722	84%	2116	44%
(UNDEFINED)	32	0%	29	83%	69	45%
CLOSED		0%	4139	93%	2005	86%
UNCACHED	0	0%	0	0%	0	0%

```
/stop-dab-caching
```

```
NDB0022 DAB CACHE BUFFER 'BUFFER4' IN CACHE MEDIUM 'M-M' HAS BEEN DELETED
NDB0022 DAB CACHE BUFFER 'BUFFER#001' IN CACHE MEDIUM 'M-M' HAS BEEN DELETED
```

Notes

1. The cache area is not released if any of its data cannot be completely saved to disk. With GS as the cache medium, message NDB0031 is output to ask whether the cache area is to be detached from the GS partition. If this is not possible (e.g. because areas of the volume are (partially) used by DMS), message NDB0057 is output. If the cache area can be detached from the GS partition, the main memory data of the cache area is deleted, while GS data is retained for subsequent reconstruction. The GS partition is detached.
2. Unconditional release of a cache area is requested by means of the `/FORCE-STOP-DAB-CACHING` command.

8 Error handling

This chapter provides information on the reporting and handling of errors that occur in connection with a cache area created by means of DAB or PFA commands in any of the cache media main memory (MM) and global storage (GS).

8.1 General information

The following error situations may arise during operation of a cache area:

- failure of the connection between server and buffered external volume; write data still in the cache cannot be written to disk.
- failure of the crypto box; write data of encrypted files which is still in the cache cannot be written to disk.
- disk error affecting the supported data areas; write data still in the cache cannot be written to disk.
- cache memory error in cache areas with dual data storage on two GS units; the cache data cannot be stored in dual mode if one GS unit is detached or failed, however it can still be accessed on the remaining GS unit. Depending on the requirement in terms of performance and data availability, caching can be kept up or continued as read caching in mono mode.
- cache memory error; no further data can be read from the cache medium; any write data still in the cache and not yet saved to disk is lost (unless duplicate data is maintained in GS).

8.2 Description of individual error situations

8.2.1 Failure of the connection between server and buffered disk

The failure is reported to the operator console by the BS2000/OSD disk error handling function.

If a cache area is used with CACHING-MODE = *READ, no action is required since all data exists both on disk and in the cache.

If cache areas are used in write or read/write mode, the error is reported by the cache handler following any of the actions listed below:

- /STOP-DAB-CACHING command for ADM-PFA cache area:


```
NDB0029 DAB CACHE BUFFER '(&00)' IN CACHE MEDIUM '(&01)' WAS NOT
COMPLETELY SAVED TO DISK STORAGE DUE TO ERROR (&02)
NDB0023 DAB CACHE BUFFER '(&00)' IN CACHE MEDIUM '(&01)' WAS NOT DELETED
```
- /EXPORT-PUBSET command for pubset with ADM-PFA cache area:


```
NDB0048 CACHE DATA OF VOLUME WITH VSN '(&00)' COULD NOT BE SAVED
COMPLETELY TO DISK STORAGE
```
- /STOP-PUBSET-CACHING command for PFA cache area:


```
DMS1364 UNABLE TO SAVE CONTENTS OF CACHE BUFFER IN MEDIUM '(&00)' FOR
PUBSET OR VOLUME SET '(&01)'
```
- /EXPORT-PUBSET command for pubset with PFA cache area:


```
DMS1364 UNABLE TO SAVE CONTENTS OF CACHE BUFFER IN MEDIUM '(&00)' FOR
PUBSET OR VOLUME SET '(&01)'
```
- closing a file of a buffered PFA pubset:


```
DMS0E27 ERROR WHEN CLOSING FILE. I/O OPERATION TERMINATED DUE TO HARDWARE
ERROR
```

Any files not saved to disk can be identified for PFA caching via the command

```
/SHOW-FILE-ATTRIBUTES :pvsid:*,SELECT=*BY-ATTRIBUTES(
STATUS=*PARAMETERS(CACHE-NOT-MAVED=*YES))
```

All cache data of a cache area is written to the associated disks with the exception of the data affected by the hardware problem so that access to the corresponding disk is not possible. The data areas affected can be determined by ADM-PFA or (USER-)PFA via the /SHOW-DAB-CACHING command (independently of caching via ADM-PFA or USER-PFA). If caching is performed via ADM-PFA and AREA=*FILE or via USER-PFA, the files are marked by the PINNED DATA status flag.

Response

- Return the disks to an accessible state (have the disk subsystem checked by a service technician, if necessary), then save cache data to disk by initiating the release of the cache area (again).
- In the event of GS operation in parallel HIPLEX mode, the cache areas can be “reattached” to another HIPLEX node (see [section “Reconfiguration in the computer network \(parallel HIPLEX\)” on page 53](#)).

8.2.2 Disk error affecting the supported data areas

The information given in the preceding subsection about error reporting (see [page 146](#)) also applies to this type of error.

Since read access to the data buffered in the cache areas is still possible, it may still be possible to save the affected files.

Response

Where individual files are cached, these can be saved by copying them to another disk (`/COPY-FILE` command). If PFA caching is used, the destination file must be created explicitly prior to copying (`/CREATE-FILE` command). This is because closing the source file results in a `CLOSE` error (`DMS0E27`); an implicitly created destination file would be deleted during copying.

Defective files must subsequently be deleted to permit the cache area to be released. Otherwise, release of the cache area must be forced by means of the commands `/FORCE-STOP-DAB-CACHING` and `/FORCE-DESTROY-CACHE`.

8.2.3 Detachment/failure of a GS unit with dual data storage

With dual data storage in GS no data will be lost if a GS unit is detached or fails, i.e. the connection to the host / cache memory fails (with mono data storage, on the other hand, data will be lost if a cache memory defect occurs). Data security will, however, be reduced as caching can only be continued in mono mode, i.e. data in the cache may be lost if the second GS unit in a dual partition fails also.

DAB's handling of such error states (dual GS errors) can be influenced by means of the ON-DUAL-GS-ERROR parameter in the subsystem initialization file SYSSSI.DAB.090 (the modification of the subsystem initialization file is permitted in this case). Systems support can program options appropriate to local requirements for optimized performance, increased data security right through to disaster recovery concepts (in connection with mirrored disks).

Description of the parameter values

The ON-DUAL-GS-ERROR parameter can assume the value BY-OPERATOR, CONTINUE-ACCESS, CONTINUE-CACHING-MODE-READ, STOP or HOLD-GS-ACCESS; the default is BY-OPERATOR. The meaning of the dual GS errors and DAB's response to them are explained below.

ON-DUAL-ACCESS-ERROR=BY-OPERATOR

In the event of a dual GS error, a message will appear on the console asking the operator how caching for the cache area in question is to be continued:

```
NDB0131 WARNING: THE OPERATING OF CACHE BUFFER '(&00)' IS RESTRICTED TO GS-UNIT '(&01)' DUE TO GS-FAILURE. HOW SHOULD THE CACHING SERVICE BE CONTINUED?
```

The operator can respond to this message as follows:

- C** corresponding to the parameter value CONTINUE-GS-ACCESS:
Caching is continued in mono mode on the remaining GS unit with reduced data availability in comparison with dual caching but still with maximum performance.
This error state is indicated by the following message on the console:

```
NDB0054 WARNING: THE OPERATING OF CACHE BUFFER '(&00)' IS RESTRICTED TO GS-UNIT '(&01)' DUE TO GS-FAILURE.
```

- R** corresponding to the parameter value CONTINUE-CACHING-MODE-READ:
Caching is continued in read mode and all write data in the cache is automatically written to the relevant disk. Full data availability is guaranteed, however data throughput will be reduced due to the lack of write caching.

This error state is indicated by the following messages on the console:

```
NDB0132 CACHE BUFFER'(&00)' WILL SERVICE READ-DATA OPERATIONS ONLY DUE TO
DUAL-GS ACCESS FAILURE.
NDB0134 SAVE OF CACHE DATA FOR CACHE BUFFER '(&00)' HAS BEEN STARTED
NDB0135 SAVE OF CACHE DATA FOR CACHE BUFFER '(&00)' HAS BEEN TERMINATED
SUCCESSFULLY
```

- S** corresponding to the parameter value STOP:
The cache area will be released. Data availability is guaranteed here also, however the complete absence of caching will mean an inevitable loss of performance.
This error state is indicated by the following messages on the console:

```
NDB0134 SAVE OF CACHE DATA FOR CACHE BUFFER '(&00)' HAS BEEN STARTED
NDB0022 DAB CACHE BUFFER '(&00)' IN CACHE MEDIUM '(&01)' HAS BEEN DELETED
```

ON-DUAL-GS-ERROR=CONTINUE-GS-ACCESS / CONTINUE-CACHING-MODE-READ / STOP

The parameter values CONTINUE-GS-ACCESS, CONTINUE-CACHING-MODE-READ and STOP each specify a particular reaction to dual GS errors. The reactions initiated correspond to the responses C, R and S to the error message NDB0131 (see the parameter value QUESTION), however they are executed without the operator's further involvement.

ON-DUAL-GS-ERROR=HOLD-GS-ACCESS

Domino mode specifies a particular reaction to dual GS errors. This reaction is triggered without the operator's further involvement.

Following a dual GS error, all further attempts to access the cache area in question for writing will be suppressed by means of I/O errors in order to keep the data consistent on both GS units.

Domino mode permits the formulation of a disaster recovery concept based on the use of two servers with access to the same GS (with two GS units) and locally mirrored disk storage systems. This mode is only useful on a disk configuration made up of disk storage systems with local mirroring.

In the event of a GS unit failure (or the failure of the connection to the GS unit), the equality of the data on the GS units will be retained so that when a failover is executed to the backup server, the most recent data up until that point can still be accessed. All GS dual partitions will be equalized on the backup host as part of this failover so that any data differences on the GS units arising from write access attempts during the failure can be reconciled.

The advantage of this configuration (and the GS error handling setting) is that, in the event of the failure of a server, GS unit or disk subsystem, a changeover to the backup server can be effected without any data being lost (even if using write caching in GS).

The disadvantage is that the availability of the individual systems is reduced: A changeover to the backup host is unavoidable in the event of a failure on a GS unit, even if the remaining GS unit can still be used.

Greater clarification of the disaster prevention requirement and concept is needed (if necessary in conjunction with consultants from the Competence Center) before this parameter can be used.

Following recovery of the memory error

Once the memory error has been recovered, the defective GS unit can be reattached using the `/ATTACH-GS-UNIT` command. The differences in the data on both memory areas in a partition (with `CONTINUE-GS-ACCESS/CONTINUE-CACHING-MODE-READ/STOP`) following the failure of the reattached GS unit will be eliminated through the equalization of the memory areas. Once all of the partitions in question have been successfully equalized, caching will be continued in the original mode, i.e. the mode in use prior to the memory error, e.g. full dual caching will be recommenced and any cache areas switched to read caching will be switched back to write caching.

8.2.4 Cache memory errors with mono data storage

With mono data storage, data cannot be read from the cache medium following a cache memory error and any write data in the cache will be lost unless it has been written to disk. This scenario can occur with cache areas in the main memory and GS cache media:

- Cache area in main memory:
If a memory error occurs when accessing a main memory cache segment, the data access operation in question will, if possible, be redirected to the relevant external data memory. An I/O error will be generated if the data access operation cannot be redirected, for example because the only up-to-date version of the data to be accessed is in the cache memory.
- Cache area in GS:
This defect relates to a memory error or the failure of the connection to GS which completely prevents the execution of data access operations on GS. This includes all access errors in mono partitions and access errors in dual partitions if neither GS unit in a partition of this type can be accessed.

DAB indicates errors of this type by means of messages on the console. These messages are of the type “asynchronous question” and are output for each cache area for which an error was found.

DAB automatically starts to write cache data to disk for areas where an error was detected. The affected area will be released once the cache data has been successfully written to disk. Even if the area cannot be released, it remains locked against further caching of data.

Message indicating an access error:

```
NDB0043 THE FOLLOWING CACHE BUFFER CONTAINS DEFECTIVE BLOCKS: '(&00)'
```

No data will be lost when using cache areas with read caching as all data will be on both the disk and in the cache. Cache areas can be released using the commands `/STOP-DAB-CACHING` and `/STOP-PUBSET-CACHING`.

When using cache areas with write or read/write caching, the cache may contain defective blocks for which there is no equivalent on disk. In this case, the data in the cache can no longer be read and the blocks are lost. The affected files can only be reconstructed if a data backup is available, and then only to the version saved in that backup. When caching entire volumes in write mode, the affected volume may need to be reinitialized. The affected data area can be found using the command `/SHOW-DAB-CACHING` for the relevant cache area. When caching using ADM-PFA and `AREA=*FILE` or in the case of user PFA, files are flagged by the status `PINNED DATA`.

8.3 GS caching with DAB and disaster recovery

This section will show how the use of GS as a cache medium and the DAB cache handler can be integrated into the disaster recovery concept for BS2000/OSD.

The following configuration is to be used as the basis for the description:

- 2 nodes (1 active node, 1 standby system),
- 2 Symmetrix storage systems, cross-cabled and linked to each other via SRDF,
- 2 GS units, both attached to both systems

For the purposes of optimum data availability caching is to be carried out in dual mode.

In order to ensure that, in the event of a disaster occurring, the data can be transferred from one node to the other without loss, it is necessary that the data on both Symmetrix storage systems and on both GS units is maintained at the same status. This prerequisite is fulfilled as long as both GS units can be accessed from the active node and that the SRDF connection between the Symmetrix storage systems functions correctly. However, the data on the two Symmetrix systems is no longer consistent if the dual data storage can no longer be guaranteed. We shall consider the following situations:

Situation 1: None of the GS units can be accessed

Since DAB acknowledges all write accesses with I/O errors and the affected cache areas can probably not be released, the user must migrate the affected data applications to the standby system.

Situation 2: Only one of the GS units can be accessed

Since data can only be stored to one of the units of the dual GS partition, a disaster occurs may lead to a critical situation. If it is no longer possible to access the GS unit with the current data from the standby system, then it will not be possible to migrate the applications to the standby system without loss of data. It is important to make sure that it is not possible to have a different data statuses on the two GS units, since it cannot be known in advance which of the GS units is actually able to access the standby system. DAB allows you to activate domino mode for dual GS accesses. This ensures that for every write I/O operation affected by a dual GS access error, an I/O error is generated. This I/O error indicates to the affected applications that the data access was not carried out successfully. In a situation like this, the user is then able to migrate their data applications to the standby system without loss of data and then restart the application.

To ensure a comprehensive disaster recovery, a configuration of this nature must have domino mode set for the operation of mirrored Symmetrix storage systems as well. Domino mode can be activated using the parameter assignment `ON-DUAL-GS-ERROR = (*)HOLD-GS-ACCESS` in the subsystem parameter file or in the `/MODIFY-DAB-PARAMETERS` command.

To ensure optimum preparation for the event of a disaster it is also sensible to make sure that caching with dual GS is only run under domino mode circumstances. The behavior of DAB when domino mode is activated may reduce the overall availability of the system, for example, a DUAL-GS error may occur but it is not possible/or not sensible to migrate to the standby system for other reasons. The user can then use the `/MODIFY-DAB-PARAMETERS` command to switch from domino mode to mono mode (`*CONTINUE-GS-ACCESS`) or to read caching (`*CONTINUE-CACHING-MODE-READ`) in order to continue working locally.

Abbreviations and glossary

This chapter provides an alphabetical list, accompanied by explanations, of the abbreviations and technical terms used in this manual.

ADM-PFA caching

Administrator-controlled Performant File Access caching.
Part of the BS2000/OSD HIPERFILE concept.

cache ID

Identifier of a cache area.

The identifier of an ADM-PFA cache area is defined either by systems support via the CACHE-ID operand of the START-DAB-CACHING command, or assigned internally by DAB if systems support does not define a cache ID.

The cache ID of a PFA cache area is identical with the Pubset ID (Catalog ID/Volset ID).

DAB

Disk Access Buffer

Disaster Recovery

A facility which makes it possible to restore the IT system of a company in the event of a disaster.

DMS

Data Management System

Domino mode

Special functional mode of DAB for dual GS accesses. In the event of a dual GS access errors an I/O error is generated for each write I/O operation. These errors signal to the user program that the data access was not carried out successfully and thus ensures that the migration of data applications to a standby system is carried out on-time and without loss.

DRV

Dual Recording by Volume

The DRV subsystem is a software solution for RAID1 functionality. It records all blocks of a disk on a second disk. The duplication takes place within the BS2000/OSD I/O system.

DSSM

Dynamic Subsystem Management

FIFO

First In - First Out

Strategy for handling a queue.

GS

Global Storage (cache medium)

GSMAN

Global Storage Manager

GSVOL	GS Volume Emulation GSVOL permits disks with high-performance I/O handling to be emulated in global storage (so-called GS volumes).
HIPERFILE concept	High Performant File BS2000 concept with the aim of achieving high-performance file processing and thus improving the performance of the entire system.
HIPLEX	Highly Integrated System Complex
I/O	I nput/ O utput
LMS	L ibrary M aintenance S ystem
LRU	L east R ecently U sed Selection and replacement strategy. When new data is stored, the blocks selected for replacement are those which have been least recently used.
locality	A good locality of the data ensures a high hit rate for caching according to the LRU principle. A distinction is made between “temporal locality” and “spatial locality”. “Temporal locality” implies that more often than not, the data referenced and therefore cached last will be accessed again. “Spatial locality”, on the other hand, implies that more often than not, the data spatially adjacent to the referenced data will be accessed next. This requires the prefetching mechanism to be enabled in order to increase the probability that the cache already contains the spatially adjacent data accessed next.
MM	M ain M emory (cache medium)
MSCF	M ultiple S ystem C ontrol F acility
NDM	N ucleus D evice M anagement Device management in BS2000/OSD.
operator console	Workplace at which the operator carries out his or her tasks.
parallel HIPLEX	HIPLEX whose servers share a global storage (GS)
PFA	P erformant F ile A ccess Part of the BS2000/OSD HIPERFILE concept (see user PFA caching); embedding of HIPERFILES in DMS.
pubset	P ublic volume set Common set of disks available to all users.
PVS	P ublic V olume S et see “pubset”.
PVS ID	Identifier of a pubset. In path names, it is specified in the form :catid:

SDF	S ystem D ialog F acility
SF pubset	S ingle- F eature pubset
Shared GS	see GSMAN
SM pubset	S ystem- M anaged pubset
SPD	S hared P rivate D isk Private disk made available to other users.
SM2	S oftware M onitor 2
SRDF	S ymmetrix R emote D ata F acility
SRV	S ingle R ecording by V olume A data management method which stores only one copy of the data on a physical disk (see "DRV").
SVL	S tandard V olume L abel Includes the VSN.
user PFA caching	User-controlled Performant File Access caching. Part of the BS2000/OSD HIPERFILE concept.
VSN	V olume S erial N umber
XCS network	C ross C oupled S ystem

Related publications

You will find the manuals on the internet at <http://manuals.ts.fujitsu.com>. You can order manuals which are also available in printed form at <http://manualshop.ts.fujitsu.com>.

- [1] **BS2000/OSD-BC
Executive Macros**
User Guide
- [2] **BS2000/OSD-BC
Introductory Guide to Systems Support**
User Guide
- [3] **HIPLEX MSCF (BS2000/OSD)
BS2000 Processor Networks**
User Guide
- [4] **BS2000/OSD-BC
Commands**
User Guide
- [5] **BS2000/OSD-BC
Performance Handbook**
User Guide
- [6] **SHC-OSD (BS2000/OSD)
Storage Management for BS2000/OSD**
User Guide
- [7] **openSM2 (BS2000/OSD)
Software Monitor**
User Guide
- [8] **VM2000 (BS2000/OSD)
Virtual Machine System**
User Guide

Index

A

- accelerating write access 61
- ADM-PFA caching 14, 15
 - create cache area 44, 123
 - for shared disks 37
 - force release of cache area 86
 - modify cache size (command) 89
 - modify configuration parameters 47
 - operating parameters 15
 - release cache area 44, 139
- assign data areas 123
- AutoDAB 13
- automatic caching 34

B

- BS2000 system shutdown 84
- buffering, home pubset 16

C

- cache area
 - create 123
 - for exclusively allocated disks 54
 - for shared imported pubset 56
 - force release 86
 - modify configuration parameters 47
 - monitor 102
 - release 139
 - release at shutdown 46
 - restoration in GS 51
- cache handler 7
- cache media 14, 21
 - CPU requirements 73
 - of DAB 21
- cache segment size, modify 47
- cache size, expand or reduce 47

cacheing

- automatic 34
- dynamic reconfiguration 47
- cacheing modes 19
 - define 123
 - modify 48
- cacheing techniques, define 123
- commands 85
 - for GS operation 29
 - FORCE-STOP-DAB-CACHING 86
 - MODIFY-DAB-CACHING 89
 - overview 85
 - SHOW-DAB-CACHING 102
 - START-DAB-CACHING 123
 - STOP-DAB-CACHING 139
- CPU requirements of cache media 73
- creating
 - DAB cache area 44
 - PFA cache area 45
- creating and releasing partitions 29

D

DAB 155

- ADM-PFA and USER-PFA caches 66
- and DRV 67
- and GSVOL 67
- and multicomputer operation 37
- and POSIX 68
- cache area, creating 44
- cache area, force release 86
- cache area, release 44
- cache area, restoration in GS 51
- cache media 21
- caching techniques 33
- commands, overview 85
- delivery units 81
- display configuration 102
- efficient use 60
- notes on usage 64
- partitions, create and release 29
- performance behavior 70
- reducing I/O system load 73
- start 84
- stop 84
- terminate 44
- unloading during live operation 84

data area, adapt 47

disaster recovery 152

disconnection of cache areas 54

disk error 147

display information about DAB configuration 102

domino mode 100, 152

DRV and DAB 67

DSSM 81

E

efficient use 60

error handling 145

F

failure of the connection between server and buffered disk 146

file encryption 63

FORCE-OUT operand 73

- modify setting 48

FORCE-STOP-DAB-CACHING command 86

frequent read access 60

G

global storage 23

- commands for operation 29
- DAB partition 25, 29
- global utilization 42
- global utilization (HIPLEX) 27
- GS unit 23, 25
- GS users 24
- local utilization 27, 41
- partition ID 24
- partitions 24
- real 28
- restoring DAB cache areas 51
- sample configuration 29
- server crash 54, 56
- storage organization 25
- structure 24
- system/disk connection failure 54, 56
- utilization levels 24
- virtual 28

GS (global storage) 23

GS partition

- reattachment unit 57
- virtual GS under VM2000 28

GSMAN 23

GSVOL and DAB 67

H

HIPERFILE concept 14

home pubset, caching 64

I

I/O time 70
improving I/O time 70
intelligent caching 13

M

main memory (MM) 22
 attributes 22
measuring the response time 70
MM (main memory) 22
MODIFY-DAB-CACHING command 89
multicomputer operation 37

P

paging area, buffering 64
parallel HIPLEX 23, 53
parameters 89
 modify dynamically for a DAB cache area see
 MODIFY-DAB-CACHING 89
partition, create and release 29
performance behavior 70
PFA cache area, create 45
PFA caching
 create cache area 44, 45
 for shared disks 39
 modify configuration parameters 47
 release cache area 44, 45

R

read access, frequent 60
read cache 19
 reducing channel load 73
 response time 73
read caching 19, 60
read/write cache 19
 response time 73
read/write caching 20, 62
Readme file 10
reattachment unit 57
reconfiguration 47
 in the computer network (GS) 53
reconstruction of DAB cache areas in GS 51
reducing channel load, read cache 73
releasing
 DAB cache area 44
 PFA cache area 45
response time
 with read cache 73
 with read/write cache 73
restoration of DAB cache areas in GS 51

S

saving, FORCE-OUT operand 73
sequential file processing 60
shared disks 37
Shared GS 157
shared private disk 37
shared subset 37
SHOW-DAB-CACHING command 102
shutdown of BS2000 84
SM2 monitoring report 74
SoftBooks 11
software caching 14
START-DAB-CACHING command 123
START-FILE-CACHING command 16
STOP-DAB-CACHING command 139
storage media of DAB 21
storage organization
 logical 25
 physical 25

T

terminating DAB [44](#)

threshold for buffering (FORCE-OUT) [48](#)

U

USER-PFA caching [14](#), [15](#)

V

VM2000

 real GS [28](#)

 virtual GS [28](#)

W

write access, accelerate [61](#)

write cache [19](#)

write caching [20](#), [61](#)

 notes [64](#)