English

FUJITSU Software BS2000

DAB V9.5

Disk Access Buffer

User Guide

Edition July 2017

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

Contents

1 Preface

The DAB subsystem is the central BS2000 cache handler for the main memory 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), providing information services for status inquiries, and handling I/Os with the cache. 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
- 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`, `/SHOW-DAB-CACHING` und `/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 V9.5 can run under BS2000 OSD/BC V11.0 and higher on the supported servers.

1.1 Objectives and target groups of this manual

This manual is intended for BS2000 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 describes the DAB software product in the following chapters:

chapter "DAB caching"
: provides information about automatic caching, on the function of DAB within the HIPERFILE concept, about caching modes and methods and about the cache medium main memory (MM).

chapter "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, and dynamic reconfiguration changes during caching.

chapter "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 "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 "Commands"
: contains an overview of the DAB commands, followed by detailed descriptions of the individual commands.

chapter "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.

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

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:

- The manual has been adapted to BS2000 OSD/BC V11.0.
- Global Storgage is no longer supported as a cache medium. The relevant sections in this manual have been deleted. The parameters in the DAB commands relating to global storage are no longer visible in guided dialog. Their description in this manual has been deleted.
- New section “Throughput optimization with AutoDAB”.
- The MODIFY-DAB-PARAMETERS command has been deleted.

1.4 Notational conventions

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

- **BS2000 servers** for the servers with /390 architecture and the servers with x86 architecture. These servers are operated with the corresponding BS2000 operating system.
- **/390 servers** for the Server Unit /390 of the FUJITSU Server BS2000 SE Series and the Business Servers of the S Series
- **x86 servers** for the Server Unit x86 of the FUJITSU Server BS2000 SE Series
- **SE servers** for the FUJITSU Server BS2000 SE Series (Server Units /390 and x86)
- **S servers** for the S 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), DAB caching modes (read cache, write cache and read/write cache) and the main memory cache medium.

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 22.

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 HIPERFILE (**hi**gh-**per**formance **file** access) concept offers caching in the main memory cache medium (MM) both via the command interface of the subsystem involved (ADM-PFA caching) and via a uniform command interface integrated in DMS (user PFA caching). ADM-PFA caching and user PFA caching are explained in more detail on page 13.

The HIPERFILE concept is supported by the DAB subsystem. 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.

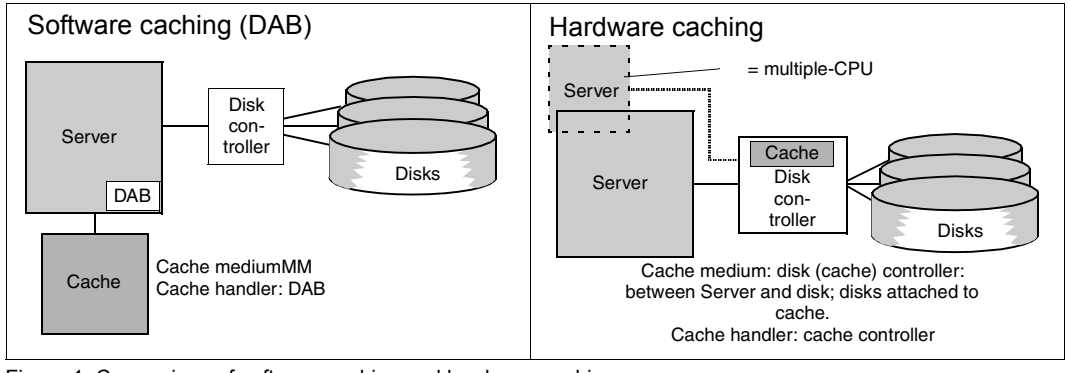


Figure 1: Comparison of software caching and hardware caching

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 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 “Introduction to System Administration” [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 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` must be noted.

The home pubset cannot be buffered in any of the cache media managed by DAB with PFA caching. 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
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 (**hi**gh-**per**formance **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] manuals):

```
/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

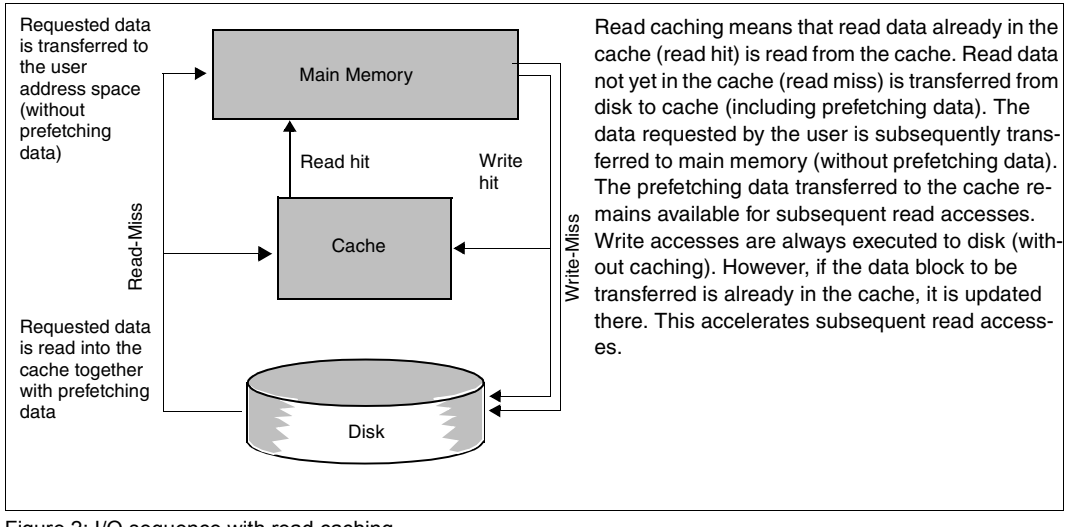
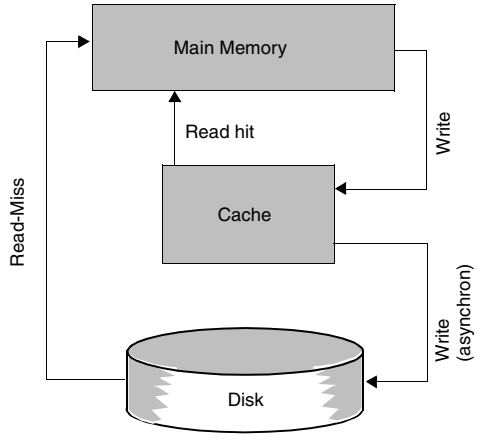


Figure 2: I/O sequence with read caching

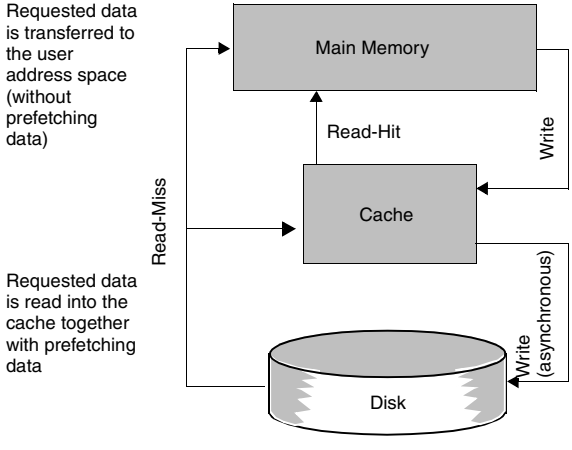
Write caching



In the event of a read hit, accesses are executed directly from the cache. In the event of a read miss, the data is read from disk. In contrast to read or read/write caching, it is, however, not entered in the cache.
Write accesses are always executed directly in the cache; this applies both to data blocks already present in the cache, which are overwritten, and data newly entered in the cache. When the entire cache area is filled with data, the data that has been accessed least recently and has already been saved to disk is overwritten. For the application, the write job is completed as soon as the data has been entered in the cache.
DAB performs threshold-controlled save runs to ensure that the data accessed least recently is saved to disk. In the event of the entire cache being filled with data not yet saved to disk due to abnormally frequent write accesses, the next cache miss causes the write data to be written directly to disk without caching (cache overflow). For write data with good locality, write or read/write caching reduces the number of write accesses to disk, since the user-requested write accesses are collected in the cache and transferred to disk in larger blocks.

Figure 3: I/O sequence with write caching

Read/write caching



Read jobs are handled with a read/write cache as with a read cache: in the event of a read hit, the data is read from the cache, in the event of a read miss, the data is entered in the cache and transferred to main memory.
Write jobs are handled as with a write cache: the data is directly entered in the cache and saved from cache to disk asynchronously.
For write data with good locality, write or read/write caching reduces the number of write accesses to disk, since the user-requested write accesses are collected in the cache and transferred to disk in larger blocks.

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

2.4 The Main Memory cache medium

DAB uses the main memory (MM) as cache medium.

Main memory has the following advantages compared to external memory media:

- 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, but because of other users (components of the operating system) fewer data spaces may be available. This will determine the maximum number of DAB cache areas.

The DAB subsystem permits main memory areas of any BS2000 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” on page 48.

3 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

3.1 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.

3.1.1 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 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.

3.1.2 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 modifiy 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 31).

3.2 DAB in multiprocessor operation

Shared pubsets can be supported by 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 pubsets, the possible caching variants must be looked at individually. The following table provides a basic summary of the executions:

	ADM-PFA	**USER-PFA**
Read caching	AutoDAB: system monitoring possible Manual DAB: within the responsibility of systems support	Read caching is the default. Only locally processed files profit here.
(Read/)write caching	not supported	not supported

Table 2: DAB caching of pubsets

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.

3.2.1 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.

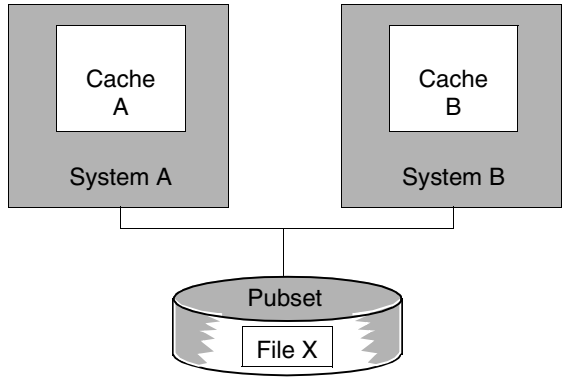
Caching shared data areas in write or read/write mode is not possible and is prevented by DAB. The SHARED-DISK-SUPPORT operand is not processed in this case and an appropriate message is output.

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

3.2.2 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 the same BS2000 version is running on all systems that have access to the shared pubset.

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.



1. File X is opened on system A.
2. File X is processed (with cache A).
3. File X is closed: cache data is written and invalidated.
4. File X is opened on system B and processed with cache B.

Figure 5: Concept for supporting shared pubsets with DAB

Caching in main memory

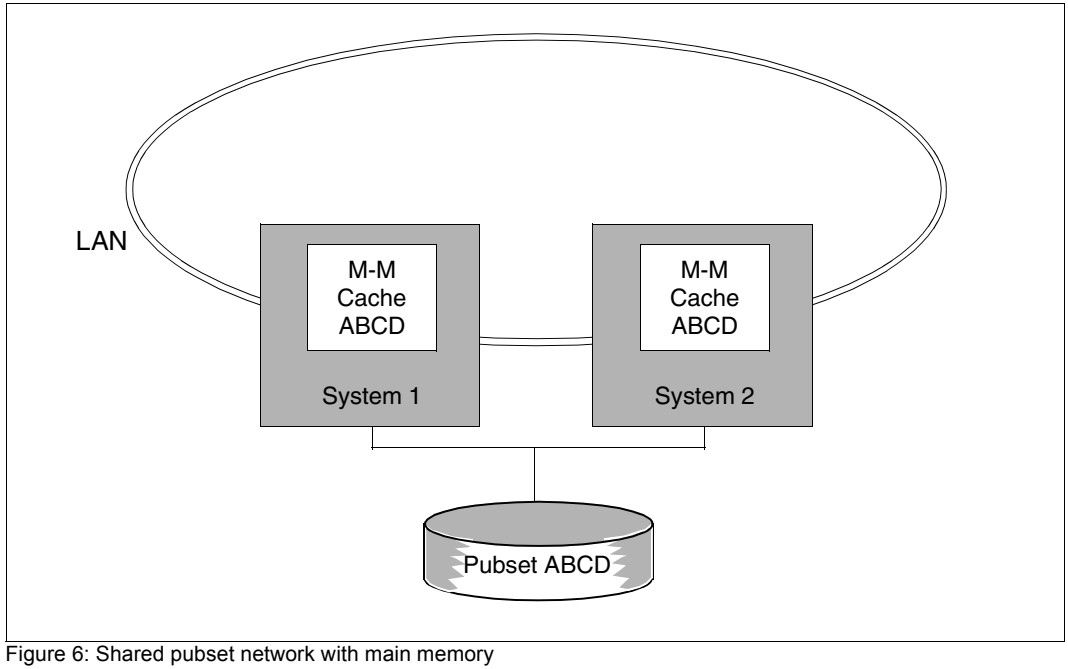


Figure 6: Shared pubset network with main memory

Main memory 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 systems.

3.3 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 an ADM-PFA cache already exists for files on the pubset or volumeset.

3.3.1 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.

3.3.2 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).

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.

3.3.3 Releasing cache areas during shutdown

When terminating the operating system (shutdown) DAB releases all existing cache areas.

3.4 Dynamic reconfiguration

3.4.1 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.

3.4.2 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 26).
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 MODIFY-DAB-CACHING AREA=*RESET-FILE(...)
ADM-PFA cache areas with AREA=*FILE (...)	MODIFY-DAB-CACHING AREA=*ADD-FILE(...)
User PFA cache areas	START-FILE-CACHING

Table 3: 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 58).

4 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 and how to relive the pressure on the I/O system.

4.1 Efficient use of DAB

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

4.1.1 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.

4.1.2 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 user can specify during cache area definition or in ongoing operation (`/START-DAB-CACHING`, `/MODIFY-DAB-CACHING`, `/MODIFY-PUBSET-CACHE-ATTRIBUTES` commands, `FORCE-OUT` operand) at which point the transfer occurs. The transfer of the buffered data from cache to disk does not normally affect program runtime.

– 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.

4.1.3 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.

4.1.4 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 folllows:

- 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.

4.2 Notes concerning DAB usage

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 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.
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, 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.

   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.
6. All assigned cache areas must be terminated before formatting disks (VOLIN utility) or regenerating pubsets (SIR 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 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. The applies for both ADM-PFA and PFA caching.
9. Refer to section “DAB in multiprocessor operation” on page 25 for notes on caching with shared pubsets.
10. Cache areas 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.
11. 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`.

4.3 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.

4.3.1 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.

4.3.2 DAB and external disk storage systems

The disk storage systems served by BS2000 (e.g. ETERNUS DX) are high availability disk storage systems which fulfill 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].

Local replication in storage systems

Local replication 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.

4.3.3 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).

4.4 Performance behavior

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

4.4.1 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:

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

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 7 on page 46). 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.

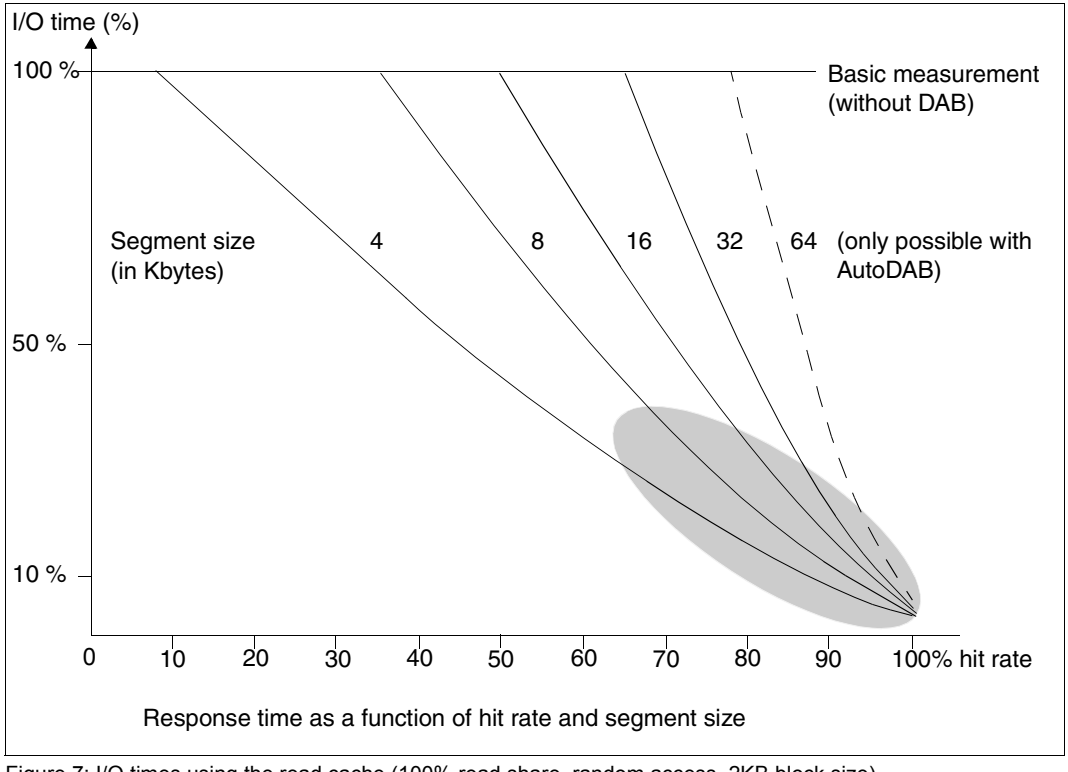


Figure 7: I/O times using the read cache (100% read share, random access, 2KB block size)

The figure 7 on page 46 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 48), 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 7 on page 46 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.

4.4.2 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.

4.4.3 CPU utilization

Additional CPU requirements are lowest with cache areas in 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 22).

4.4.4 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).

4.4.5 Throughput optimization with AutoDAB

Two measurements with the following access profiles were conducted to demonstrate the effects of AutoDAB:

Measurement 1: 8 tasks accessing with write access to one file each.
4 tasks accessing with a sequential access pattern, 4 with a random access pattern.
There are no write I/Os.

Measurement 2: 8 tasks accessing with random access pattern and 25% write share.

The measurements were taken on an S server. File size was 500 MB and the files were located on two NK2 volumes. I/O length was 4 KB and the duration of measurement was 5 minutes each. Measurements were performed without and with AutoDAB cache. AutoDAB cache in CACHING-MODE=*READ has a cache size of 2,000 MB in the main memory.

The measurement results are shown in the following table:

Measure-ment	Access type	IO/s without DAB cache	MB/s without DAB cache	IO/s with DAB cache	MB/s with DAB cache
1	sequential	32,809	128	58,106	226
1	random	32,935	128	78,832	307
2	random	31,018	121	79,746	311

With AutoDAB the I/O throughput has been considerably improved. This is also true for the overall throughput.

The reasons for this are:

- The better the hit rate in a DAB cache, the more CPU power can be converted into I/O throughput.
- In measurement 1 the sequentially processed files have a very high hit rate due to the high prefetching factor, i.e. data areas are cached in advance, and are kept practically resident in the DAB cache due to the low data volume.
- Measurement 1 also shows that 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.
- The random files in measurement 2 also have very high hit rates due to the intelligent algorithms of the AutoDAB.

5 Installing, starting and stopping DAB

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

5.1 Installing DAB

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

DAB V9.5 can execute under BS2000 OSD/BC V11.0 or higher..

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 55). 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 the Service.

  Exception

  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).
- 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>=095 for DAB V9.5):

`SYSLNK.DAB.<ver>`	Module library (/390 servers)
`SKMLNK.DAB.<ver>`	Module library (x86 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

5.2 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=09.5]` 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.

DAB can subsequently be (re)started using the command
`/START-SUBSYSTEM SUBSYSTEM-NAME=DAB[,VERSION=09.5]`.

- 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.

6 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	56
MODIFY-DAB-CACHING	Modify the parameters of a DAB cache area dynamically	ADM-PFA caching and PFA caching	58
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	69
START-DAB-CACHING	Create ADM-PFA DAB cache areas	ADM-PFA caching	84
STOP-DAB-CACHING	Release ADM-PFA DAB cache areas	ADM-PFA caching	99

[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>

Operands

CACHE-ID = <name 1..32>
Specifies the DAB cache area which is to be forcibly released.

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
	64	NDB0023	Cache area could not be released
	64	NDB0067	Cache area not released due to a system error

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**

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

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

***EXTEND**(...)

NEW-SIZE = <integer 1..8388608>(...)

<integer 1..8388608>(...)

DIMENSION = ***KILOBYTE** / ***MEGABYTE**

***REDUCE**(...)

NEW-SIZE = <integer 0..8388608>(...)

<integer 0..8388608>(...)

DIMENSION = ***KILOBYTE** / ***MEGABYTE**

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

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

***ADD-FILE**(...)

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

***REMOVE-FILE**(...)

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

***RESET-FILE**(...)

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

***ADD-PUBSET**(...)

PUBSET = <cat-id 1..4>

***REMOVE-PUBSET**(...)

PUBSET = <cat-id 1..4>

,**CACHING-MODE** = ***UNCHANGED** / ***READ** / ***READ-WR**ITE / ***WR**ITE / ***BY-CACHE-MEDIUM**

,**FORCE-OUT** = ***UNCHANGED** / ***AT-LOW-FILL**ING / ***AT-HIGH-FILL**ING / ***NO**

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.

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 84), 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 84), 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 file attributes (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	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	NDB0096	The modification could not be performed either because of a memory bottleneck, 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	NDB0100	Pubset already being served
	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

(SC2)	SC1	Maincode	Meaning
	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	NDB0124	The new cache segment size is the same as the old cache segment size
	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
	64	NDB0159	File cannot be added to the cache area owing to lack of main memory
	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

(SC2)	SC1	Maincode	Meaning
	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
	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.

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. 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 73).

Format

SHOW-DAB-CACHING	Alias: SHDABC
CACHE-ID = ***ALL** / <alphanum-name 1..32>	
,**INF**ORMATION = ***STD** / ***SUMM**ARY / ***SYSTEM-CACHED-FILES**(...)	
***SYSTEM-CACHED-FILES**(...)	
CACHING = ***ANY** / ***ACTIVE / *SUSPENDED**	

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.

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	NDB0024	No cache area currently active
	64	NDB0068	Cache area not found
	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).
- Specified (desired) size of the cache area
  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 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).

  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 pubsets (with AREA=*FILE)
  (definition whether read caching of a data area is to be performed even if the associated disk is operated in 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 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).

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

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	var(*LIST).CACHE-MED	S	*MAIN-MEM	
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
Status of the served file *Y=file or partial qualification currently served by DAB *Y(SHARED)=file on shared shared disk served *Y(STOP)=cache area just released *Y(PARTIAL)=file partly served, as it is distributed over multiple disks, not all of which are exclusively allocated *Y(MANDATORY)= file or partial qualification whose caching is explicitly specified with /MODIFY-DAB-CACHING. *Y(PINNED-DATA)= *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 *N(INVALID)=data description of DAB and of the disk space allocator no longer match because of errors *N(SHARED)=file on shared disk not served *N(CLOSE)=file is closed *N(DELETED)=file being deleted *N(SUSPENDED)=file excluded from caching because of poor cache utilization *N(CLOSE)=file is closed *N(DELETED)=file being deleted *N(SUSPENDED)=file excluded from caching because of poor cache utilization *N(REMOVED)=file or partial qualification explicitly excluded from caching with /MODIFY-DAB-CACHING *N=areas not being served by DAB at present *UNDEF=status could not be determined ...	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

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	
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
Number of write accesses	var(*LIST).NUM-OF-WRITE	I	<integer 0..4294967296>	3c
Status of cached private disks	var(*LIST).PRIV(*LIST).SERVICE	S	*N *Y *UNDEF	1d

Output information	Name of the S variable	T	Contents	Condition
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) *UNDEFR	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
Read hit rate	var(*LIST).READ-HIT-RATIO	I	<integer 0..100>	3c
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=*FILE) *TRUE=information is output *FALSE=information is not output	var(*LIST).SHARED-DISK-SUP	B	*UNDEF *TRUE *FALSE	

Output information	Name of the S variable	T	Contents	Condition
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
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 = M-M
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:$TSOS.ARCHIVE.SAVE.FILE.<date>.<time>.WORK02
       1-1404        0%     N(CLOSED)
:B304:$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:$DB125SQL.SYSMSGA.ESQL.011
       1-30          0%        Y             159     95%           0
:B304:$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 = M-M
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=c#priv),struc-out=lvar3

CACHE-ID     = C#PRIV
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>
------------------------------------------------------------------------------
CATID/VSN    TYPE           SERVICE
=========    ====           =======
WK51EB     PRIVATE-DISK        Y
WK51E1     PRIVATE-DISK   N(SHARED)
------------------------------------------------------------------------------
FILE         AREA           SERVICE   # OF READS  RD-HIT # OF WRITES  WR-HIT
====         ====           =======   ================== ==================
NO DATA AREAS

/show-var lvar3
LVAR3(*LIST).CACHE-ID = C#PRIV
LVAR3(*LIST).CACHE-MED = *MAIN-MEM
LVAR3(*LIST).CACHE-SEGMENT-SIZE = *UNDEF
LVAR3(*LIST).CACHE-SIZE = 20
LVAR3(*LIST).SIZE-DIM = *MB
LVAR3(*LIST).SIZE-BY-FILE = *UNDEF
LVAR3(*LIST).CACHE-SIZE-ACTUAL = 20
LVAR3(*LIST).SIZE-DIM-ACTUAL = *MB
LVAR3(*LIST).CACHING-MODE = *READ-WRITE
LVAR3(*LIST).FORCE-OUT = *AT-LOW-FILL
LVAR3(*LIST).SHARED-DISK-SUP = *UNDEF
```

```
LVAR3(*LIST).STATIS(*LIST).STA = *ACTIVE
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-DATE = <date>
LVAR3(*LIST).STATIS(*LIST).DATA-RESET-TIME = <time>
LVAR3(*LIST).AREA-TYPE = *BY-SYSTEM
LVAR3(*LIST).PRIV(*LIST).VOL = WK51EB
LVAR3(*LIST).PRIV(*LIST).SERVICE = *Y
LVAR3(*LIST).PRIV(*LIST).VOL = WK51E1
LVAR3(*LIST).PRIV(*LIST).SERVICE = *N(SHARED)
LVAR3(*LIST).UNCACHED(*LIST).NUM-OF-FILE = 0
LVAR3(*LIST).UNCACHED(*LIST).NUM-OF-READ = 0
LVAR3(*LIST).UNCACHED(*LIST).NUM-OF-WRITE = 0
```

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) 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 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 58).

Format

START-DAB-CACHING Alias: **SRDABC**

AREA = **<u>*BY-SYSTEM</u>**(...) / ***FILE**(...)

<u>*BY-SYSTEM</u>(...)

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

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

***FILE**(...)

FILE-**AREA** = ***NO** / list-poss(16): <filename 1..54 without-vers>

,**CACHE-SIZE** = <integer 1..8388608>(...) / ***BY-FILE**

<integer 1..8388608>(...)

DIMENSION = **<u>*KILOBYTE</u>** / ***MEGABYTE**

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

,**CACHING-MODE** = **<u>*READ</u>** / ***WR**ITE / ***READ-WR**ITE / ***BY-CACHE-MEDIUM**

,**CACHE-MEDIUM** = **<u>*MAIN-MEMORY</u>** (...)

***MAIN-MEMORY**(...)

CACHE-SEGMENT-SIZE = **<u>*32</u>** / ***4** / ***8** / ***16**

,**MEMORY** = **<u>*STD</u>** / ***ABOVE-MIN-MEM-SIZE** / ***ANY** / ***BELOW-MIN-MEM-SIZE**

,**FORCE-OUT** = **<u>*AT-LOW-FILL</u>**ING / ***AT-HIGH-FILL**ING / ***NO**

,**SHARED-DISK-SUPPORT** = **<u>*NO</u>** / ***YES**

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).

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 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.

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 data area specification.

– Manual caching at file level (specification AREA=*FILE):
  The READ caching mode is set.
– Automatic caching (specification AREA=*BY-SYSTEM):
  The READ caching mode is set for permanent files and the READ-WRITE caching mode isi set for temporary files. Temporary files are saved to disk when they are closed to ensure failsafe caching.

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 95.

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:
CACHE-SIZE ≤ 7/8 * (P – 5) 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:
CACHE-SIZE ≤ 7/8 * (P – 64) 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:
CACHE-SIZE ≤ 7/8 * (P – 64) 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:
CACHE-SIZE ≤ 7/8 * (P – 64) 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.

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 97). The operand is irrelevant for cache areas with automatic caching (AREA=*BY-SYSTEM) (see “Automatic FORCE-OUT correction” on page 23).

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 pubsets are to be supported as well with AREA=*FILE (see Notes 6 on page 97 and 7 on page 98). 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 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 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) 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	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
	64	NDB0098	Pubset not imported locally
	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	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

(SC2)	SC1	Maincode	Meaning
	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 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)
```

2. Serving the TSOSCAT file catalog of the RATS pubset with MM cache medium in read mode.

   – 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
```

3. 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):

   a) 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.

   b) Import the pubsets which are to be buffered after the load has been taken over.

   c) 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.

   d) In /START-DAB-CACHING also specify the pubsets (AutoDAB) or files (FILE-AREA cache) which are to be buffered.

   e) Export the pubsets which are to be buffered after the load has been taken over.

   f) 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 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 58.

- 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 22).

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 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 system.

Non-automatic caching

Using local system caches to serve data areas extending beyond a given system (e.g. files on 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 = **<u>*ALL</u>** / <name 1..32>	

Operands

CACHE-ID = <u>*ALL</u> / <name 1..32>
Specifies the cache areas which are to be released.

CACHE-ID = <u>*ALL</u>
All cache areas existing 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.

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	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.
2. Unconditional release of a cache area is requested by means of the `/FORCE-STOP-DAB-CACHING` command.

7 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.

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.

– disk error affecting the supported data areas; write data still in the cache cannot be written to disk.

– 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).

7.1 Failure of the connection between server and buffered disk

The failure is reported to the operator console by the BS2000 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-SAVED=*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 the Service , if necessary), then save cache data to disk by initiating the release of the cache area (again).

7.2 Disk error affecting the supported data areas

The information given in the preceding subsection about error reporting (see page 104) 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`.

7.3 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.

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.

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.

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
: **Adm**inistrator-controlled **P**erformant **F**ile **A**ccess **caching**.
Part of the BS2000 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
: **D**isk **A**ccess **B**uffer

DMS
: **D**ata **M**anagement **S**ystem

DSSM
: **D**ynamic **S**ub**s**ystem **M**anagement

FIFO
: **F**irst **I**n - **F**irst **O**ut
Strategy for handling a queue.

HIPERFILE concept
: **Hi**gh **Per**formant **File**
BS2000 concept with the aim of achieving high-performance file processing and thus improving the performance of the entire system.

HIPLEX
: **H**ighly **I**ntegrated System Com**plex**

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).
Operator Console	Workplace at which the operator carries out his or her tasks.
PFA	**P**erformant **F**ile **A**ccess Part of the BS2000/OSD HIPERFILE concept (see user PFA caching); embedding of HIPERFILEs in DMS.
Pubset	**Pub**lic volume **set** (PVS): Common set of disks available to all users.
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
SM pubset	**S**ystem-**M**anaged pubset
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 HIPERFILE concept.
VSN	**V**olume **S**erial **N**umber
XCS network	Cross Coupled System

Related publications

You will find the manuals on the internet at *http://manuals.ts.fujitsu.com*. You can order printed copies of those manuals which are displayed with an order number.

[1] **BS2000 OSD/BC**
Executive Macros
User Guide

[2] **BS2000 OSD/BC**
Introduction to System Administration
User Guide

[3] **HIPLEX MSCF** (BS2000)
BS2000 Processor Networks
User Guide

[4] **BS2000 OSD/BC**
Commands
User Guide

[5] **BS2000 OSD/BC**
Performance Handbook
User Guide

[6] **SHC-OSD** (BS2000)
Storage Management for BS2000
User Guide

[7] **openSM2** (BS2000)
Software Monitor
User Guide

[8] **VM2000** (BS2000)
Virtual Machine System
User Guide

Index