THE POSSIBILITIES ARE INFINITE FUJITSU

WHITE PAPER

BS2000/OSD
DAB – Disk Access Buffer
Intelligentes Caching mit AutoDAB

Ausgabe Juni 2009

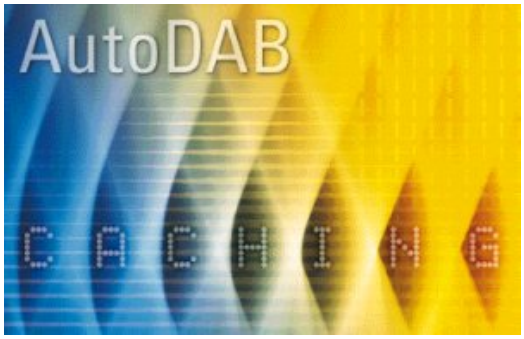
Seiten 7

To cache or not to cache? That is not the question!

Im Business Critical Computing kommt es zu hohen Performanz-anforderungen und unplanbaren Lastspitzen in den zugrunde liegenden I&C-Infrastrukturen.

Ein smartes Workload-Management und das Caching von Daten entsprechend Zugriffsprofilen, die in der Vergangenheit beobachtet wurden, sind probate Mittel, solche hohen Performanz-Anforderungen zu befriedigen und die geforderten Service-Levels einzuhalten.

Daten-Caching wird heute in allen Gliedern der I&C-Infrastrukturkette eingesetzt, also in Netzen, in Servern und in der Datenhaltung.



Inhalt

Caching 2
Daten cachen mit Raffinesse 3
Vorteile und Nutzen des DAB – Ein Beispiel 4
AutoDAB – Der autonome Cache-Manager 5
DAB im HIPLEX 5
DAB und File Encryption 6
Glossar 6

Caching

Ein häufig eingesetztes Ein-/Ausgabe-Caching-Verfahren ist Caching in Controllern in Plattensubsystemen. Die Datentransferzeit wird in diesen Fällen nicht mehr durch die relativ langsame Plattenzugriffszeit dominiert, sondern durch die Zugriffszeit auf den Halbleiterspeicher des Platten-Controllers. I/O-Transfers laufen dann mit Datenraten, die durch das Server-I/O-System und dessen Kanalstruktur vorgegeben werden.

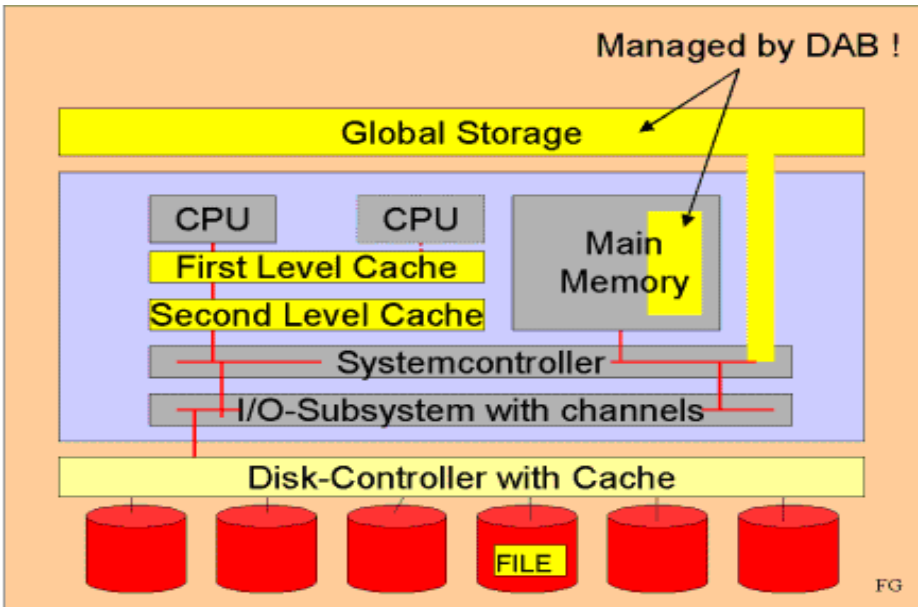
Hochentwickelte Server-Architekturen gehen jedoch noch einige Schritte weiter: Sie „cachen“ Daten in noch schnelleren CPU-nahen Halbleiterspeichern, z.B. im Hauptspeicher, in einem Global Storage oder in First / Second Level Caches. Zugriffe von Prozessoren auf diese Caches sind noch eine Größenordnung schneller, weil einerseits extrem schnelle Halbleiterbausteine zum Einsatz kommen und weil andererseits der I/O-Bottleneck samt I/O-Protokoll-Overhead umgangen wird und der Zugriff meist synchron erfolgt.

BS2000/OSD und seine Server bieten all diese Cachingmechanismen in einer Caching-Hierarchie.

Um das Caching in Hauptspeichern und Global Storages kümmert sich der BS2000-Cache-Manager DAB. DAB realisiert CPU-nahes Software-Caching, d.h. die CPUs greifen synchron auf die Cache-Speicher „Hauptspeicher“ oder/und „Global-Speicher“ zu.

Gegenüber Controller-Caching im Plattensubsystem wird der Datentransfer um ein Mehrfaches beschleunigt. DAB konfiguriert die Cache-Bereiche in den Cache-Medien, kontrolliert die Performance beim Caching, stellt Informationsdienste zu Statusabfragen bereit und realisiert die Ein-/Ausgaben zu den Cache-Medien.

Natürlich ist DAB als BS2000-Subsystem einfach zu installieren, flexibel in der Handhabung und bietet sowohl weitgehende Automatisierbarkeit als auch individuelle Optimierungsmöglichkeiten.



Daten cachen mit Raffinesse

Server lesen und speichern Daten und Instruktionen mit Hilfe einer Speicherhierarchie.

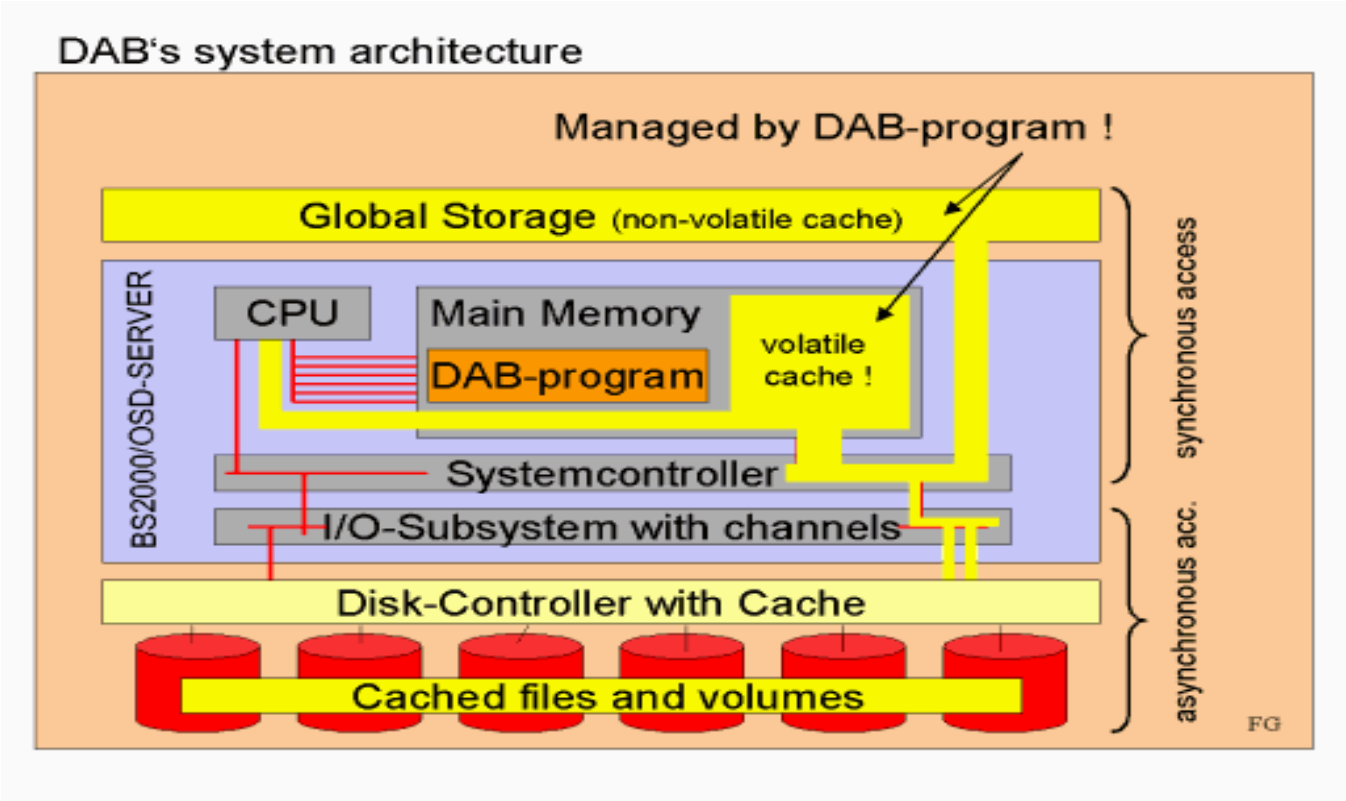
Wenn ein Programm startet, werden sukzessive Daten und Instruktionen von den relativ langsamen Platten-speichern in den Hauptspeicher gebracht. Die CPU holt sie von dort ab. Der Hauptspeicher arbeitet als Cache für die Platten. Aus CPU-Sicht ist das noch zu langsam. Daher haben moderne CPUs immer L1-Caches (Level 1) und meist auch L2-Caches (Level 2). Wenn eine CPU eine neue Instruktion ausführen will, schaut sie zuerst in den Registern nach. Sind die Daten nicht dort, wird im L1-Cache, dann im L2-Cache und dann im Hauptspeicher gesucht. Sind die Daten auch dort nicht, kommt es zum I/O-Vorgang mit Plattenzugriff. Inzwischen sind aus CPU-Sicht viele Zyklen nutzlos verronnen. Die CPU gibt auf und wendet sich dem nächsten Prozess zu. Dieser Kontextwechsel kostet wiederum zahlreiche unproduktive Zyklen.

Hier setzt in BS2000/OSD-Servern das DAB-Caching an. Es führt weitere Cache-Ebenen für synchronen CPU-Zugriff ein.

Caching ist überall

Wenn Sie sich am Morgen an Ihren Schreibtisch setzen, erwarten Sie den sofortigen Zugriff auf häufig benutzte Informationen und Arbeitsmittel. Telefon zum Beispiel, Schreibzeug und Akten gerade laufender Projekte liegen noch wie gestern und deshalb sofort nutzbar auf ihrem Schreibtisch. Der PC läuft, Tastatur und Maus sind aktiv und Sie lesen ohne Zeitverzug ihre seit gestern eingegangene e-mail. Nur in Einzelfällen gehen Sie noch zum Aktenschrank und holen seltener gebrauchte Unterlagen.

Dahinter steckt das Cache-Prinzip, nämlich das Bestreben, häufig gebrauchte Ressourcen in schnellstmöglichem Zugriff zu halten.



- DAB ist der zentrale Cache-Manager des BS2000 für die CPU-nahen Speichermedien Main Memory (lesen) und Global Storage (lesen und schreiben).
- DAB richtet in diesen Speichermedien I/O-Cache-Bereiche ein und realisiert die Algorithmen für Lese- und Schreib-Caches.
- DAB stellt über die BS2000-Oberfläche Informationsdienste zur Verfügung und arbeitet mit anderen BS2000-Komponenten eng zusammen.
- DAB-Caching kann sowohl individuell gesteuert als auch automatisiert (AutoDAB) eingesetzt werden.
- DAB kann auch mit verschlüsselten Dateien arbeiten!

Vorteile und Nutzen des DAB – Ein Beispiel

Vorteile im Vergleich zu Controller-Caching im Platten-Subsystem

- Synchroner High-Speed-Zugriff durch CPUs auf DAB-Speichermedien
- vollkommen paralleler IO-Cache-Zugriff, d.h. keine plattenspezifische Serialisierung
- automatische Auswahl der zu cachenden Daten auf Dateiebene
- dynamische Zuordnung Cache ↔ Datenbasis, entsprechend Datenflussverhalten
- DAB ist unabhängig von der Plattenspeicher-Hardware
- DAB-Medien - Größe konfigurierbar und jederzeit änderbar

Cache-Hits und Cache-Misses
„When a CPU finds data in one of its cache locations, it's called a hit; failure to find it is a miss. Every miss introduces a delay, or latency, as the processor tries a lower level. In a well-designed system with software algorithms that prefetch data before it's requested, the hit can reach 90%."

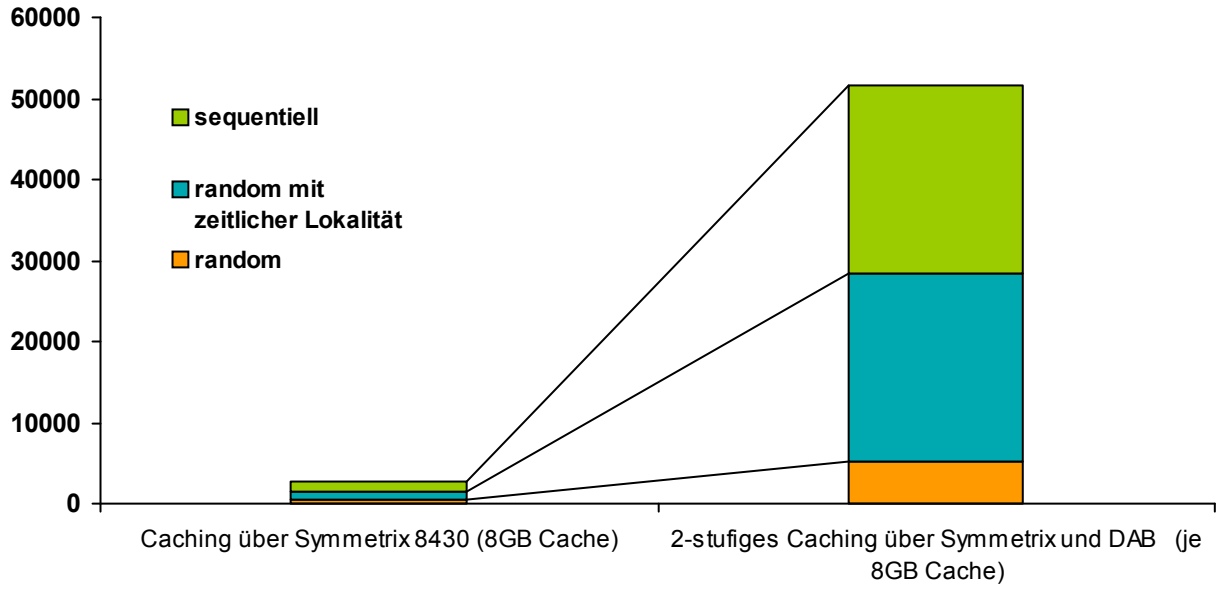
GARY H. ANTHES in CWorld

Ein Beispiel zum DAB-High-Speed-Zugriff (optimaler Fall)

Performance-Messergebnisse, die mit der Option "AutoDAB" erzielt wurden:

Anlagenkonfiguration

Server:	S170-40
Plattensubsystem:	Symmetrix 8430, 8 GB Cache, 2 Pubsets mit je 5 Platten genutzt
Global Storage:	8 GB, Nutzung durch einen Cache-Bereich



Die Grafik zeigt, wie sich der Daten-Durchsatz durch Zuschalten eines DAB-Cache-Bereichs und Anwendung von AutoDAB erhöht. Gegenüber der Verarbeitung über den Symmetrix-Cache erhöht sich die Anzahl der Datenzugriffe/sec von 2700 auf 51500 Zugriffe. ***Das entspricht einer Steigerung um den Faktor 19!***

Wie kommt diese unglaubliche Steigerung zustande?

Die enorme Hitrate im Cache kommt durch folgende Effekte zustande:

- Die **sequentiellen Dateien** haben aufgrund eines sehr großen Prefetch (Datenbereiche werden im voraus in den Cache eingelagert) eine sehr hohe Hitrate und werden dann aufgrund des Datenvolumens praktisch resident im DAB-Cache gehalten.
- Die **random Dateien mit zeitlicher Lokalität** werden aufgrund der intelligenten Algorithmen des AutoDAB ebenfalls mit sehr hohen Hitraten bedient und belegen nur einen geringen Anteil des Cache-Speichers.
- Durch den von DAB intelligent genutzten Cache-Speicher steht noch ein großer Anteil für das Caching der Dateien mit **random** Zugriffsmuster zur Verfügung, was nun ebenfalls zur Durchsatzsteigerung beiträgt.

Auch bei geringeren Hitraten als im Beispiel bringt DAB meist deutliche Performancegewinne.

AutoDAB – Der autonome Cache-Manager

Der wichtigste Aspekt des DAB-Einsatzes ist die Entscheidung, welche Anwendungen durch DAB beschleunigt werden sollen und welche Dateien oder Volumes vom DAB-Caching profitieren sollen.

DAB bietet dazu zwei Optionen:

- **Option 1 – Expertenmodus:** Hier wählt der Anwender aufgrund seiner Detailkenntnisse der Anwendungen und ihres Datenzugriffsverhaltens die zu cachende Datenbasis selbst aus.
- **Option 2 – Das sich selbst optimierende Cache-System:** Der AutoDAB, eine intelligente DAB-Komponente, wählt in diesem Modus aufgrund seiner Beobachtungen des Datenfluss- und Cache-Hit-Verhaltens die zu cachenden Dateien selbst aus.

Im einfachsten – selbst optimierenden - Fall wird der gesamte zur Verfügung stehende Cache-Speicher als ein Cache-Bereich für alle performance-relevanten Platten von der Systembetreuung konfiguriert und die Auswahl der Dateien dem AutoDAB überlassen. Er sorgt dynamisch dafür, dass häufig genutzte Dateien mit guter Cache-Hitrate gepuffert werden und Dateien mit ungünstigem Cache-Verhalten den Cache nicht nutzen.

Der AutoDAB als Cache-Experte

AutoDAB bietet eine Vielzahl von Funktionen, die intern das Caching der Dateien steuern und optimieren.

Dazu gehört als wichtigste die Erkennung sequentieller Zugriffsmuster durch permanenten Mitschnitt einer Anzahl der zuletzt durchgeführten IOs (Predictive File Table) und darauf aufbauend die Einstellung des zum Zugriffsprofil passenden Prefetch-Faktors für die ausgewählten Dateien. Um bei sequentiell bearbeiteten Dateien einen möglichst großen Prefetch-Wert zu erreichen, werden mehrere Cache-Segmente durch eine physikalische IO mit Daten gefüllt (Multi-Segment-Prefetch) und gleichzeitig vorausschauend immer ein asynchroner Prefetch angestoßen.

Die gepufferten Dateien werden bezüglich ihrer Cache-Nutzung zyklisch überwacht, d.h. die Klassifizierung in ein bestimmtes Zugriffsprofil und damit die Einstellung des Prefetch-Faktors erfolgt dynamisch. Dateien mit schlechtem Cache-Nutzungsgrad können ihren Caching-Anspruch zugunsten anderer Dateien mit höherer Hitrate und deren Anwendungen verlieren.

Schreib-IOs werden (bei Schreib-Caching) generell sehr schnell in den Cache durchgeführt, d.h. die Zugriffszeit entspricht der Zeit eines Read-Hit. Das asynchrone Zurückschreiben der Cache-Daten fasst mehrere Cache-Segmente zu einer physikalischen IO zusammen und hält so die Plattenauslastung gering..

Ein simples Rezept für den Einsatz des AutoDAB

- Wählen Sie die performance-kritischen Anwendungen aus !
- Wählen Sie die zu cachenden Datenträger mit den Daten für die Anwendungen aus !
- Stellen Sie das Cache-Medium bereit!
  – GS : Einrichten von GS-Partitionen für DAB
  – MM: Abschätzen des verfügbaren Speichers
- Richten Sie Cache-Bereiche ein und geben Sie die Caching-Methode mit *BY-CACHE-MEDIUM an!
- Kontrollieren mit: /SHOW-DAB-CACHING bzw. SM2-Reports (FILE- / DAB- / PFA- / Symmetrix-Report)

Gute Gründe sprechen für AutoDAB

- In der Regel erhebliche Steigerung des IO-Durchsatzes
- Kurze Antwortzeiten für OLTP-Anwendungen
- Drastische Reduzierung langer Batch-Laufzeiten
- Schneller Wiederanlauf von Anwendungen wie z.B. Datenbanken nach ungeplanten Ausfällen
- Wenig Expertenwissen erforderlich

DAB im HIPLEX

Zum Caching von Shared Platten (Shared Pubsets oder SPD) in einem Serververbund bietet DAB Funktionen sowohl für Lese-Caching als auch für Schreib-Caching an. Dabei handelt es sich immer um ein Caching von Dateien, die serverlokal benutzt werden (dies ist der Standard-Fall bei der Dateibearbeitung).

Lese-Caching von shared Platten wird im ADM-PFA-Konzept (DAB-Kommandos) durch AutoDAB unterstützt. Cache-Bereiche können hierfür serverlokal an den beteiligten Knoten des Verbunds eingerichtet werden. DAB stellt die Datenkonsistenz sicher, indem für die beim Schliessen gecachten Dateien die Cache-Daten invalidiert werden. So wird sichergestellt, dass nach einem zwischenzeitlichen Update der Daten von einem anderem Server aus wieder auf den aktuellen Stand zugegriffen wird.

Schreib-Caching für shared Datenträger wird von DAB ausschliesslich für Pubsets im (User-)PFA-Konzept unterstützt. Auch hierbei werden serverlokale Cache-Bereiche an den Servern des Verbunds eingerichtet.

DAB unterstützt als mögliche Konfigurationen Hauptspeicher-Cache-Bereiche an den einzelnen Servern im Verbund (dazu ist auch ein CCS-Verbund ausreichend), ebenso wie Cache-Bereiche im lokalen GS (GS nur an einer Anlage im Verbund verfügbar) und im globalen GS (GS wird von allen Verbundteilnehmern genutzt).

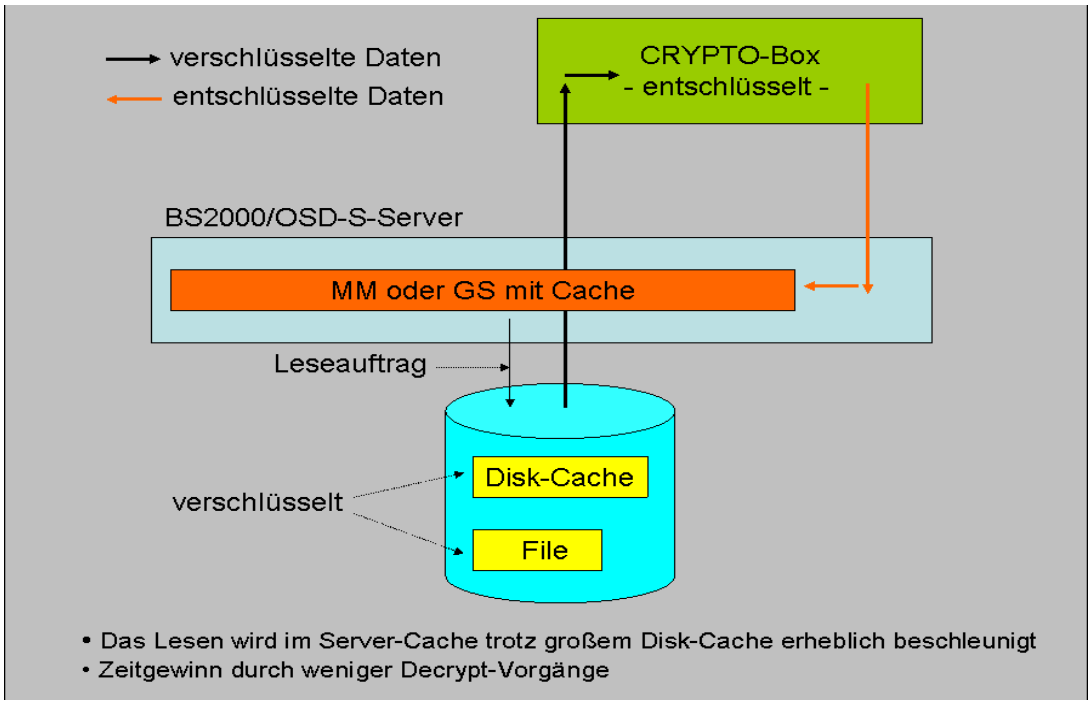
Im Falle eines global genutzten GS im HIPLEX (Parallel HIPLEX) stehen Funktionen zur Verfügung, die eine zentrale Verwaltung des GS und dessen Nutzung als Shared-Medium erlauben. Im Parallel HIPLEX liegen die DAB-Cache-Bereiche für Shared Pubsets in einer shared genutzten GS-Partition. Eine solche Konfiguration bietet Vorteile im Falle eines Serverausfalls, da DAB nachfolgend eine Rekonfiguration des betroffenen Cache-Bereichs vornimmt:

- Bei Serverausfall automatisches Umschalten der Cache-Bereiche an den Pubset-Master (bzw. Backup-Master)
- Nach Umschaltung sofortiger Zugriff auf die Cache-Daten
- Automatisches, asynchrones Zurückschreiben der Cache-Daten des umgeschalteten Cache-Bereichs auf das Pubset incl. Auflösen des (nun additiven) Cache-Bereichs

Das DAB-Caching im HIPLEX ermöglicht jedoch nicht nur eine Nutzung des GS im Verfügbarkeitsverbund. Durch DAB-Caching lässt sich ebenso ein hochperformanter Lastverbund bilden. DAB steigert die IO-Leistung des gesamten Verbunds.

DAB und File Encryption

Daten aus verschlüsselten Dateien müssen vor der Verarbeitung entschlüsselt werden. Dies erledigt die über Kanal angeschlossene Crypto-Box (S-Server-Fall) oder eine dedizierte CPU im SX-/SQ-Server-Fall. Nach der Entschlüsselung liegen die Daten im Klartext im Hauptspeicher oder in einem DAB-Cache. Natürlich kostet die Entschlüsselung Zeit, da ja z.B. im S-Server-Fall mindestens zwei weitere I/Os auf die Crypto-Box erforderlich sind. Werden entschlüsselte Daten gecacht, entfallen beim Lese-Caching bei wiederholtem Zugriff auf dieselben Daten neben den Platten-I/Os auch die zwei I/Os auf die Crypto-Box zum Entschlüsseln. Dieser Performancegewinn ist durch einen Plattencontroller-Cache nicht erzielbar, da dort die Daten immer in verschlüsselter Form vorliegen.
Das folgende Bild zeigt die Situation.



Glossar

• Begriff	• Definition
• AutoDAB	• Ein Cache-Manager, der wie ein autonomes System arbeitet, der also Caches entsprechend vorgegebener policies autonom verwalten kann. DAB (--> DAB) enthält eine solche Komponente.
• Cache	• Ein Speicher mit schnellem Zugriff, der eine Teilkopie der Daten eines anderen, in der Regel langsameren Speichers enthält.
• Cached File	• Eine Datei, die durch einen Administrator oder durch AutoDAB zum Cachen ausgewählt wurde.
• Cache-Hierarchie	• Eine Reihe hintereinander gelagerter Caches, die sich in der Regel hinsichtlich Zugriffsgeschwindigkeit und Cache-Kosten monoton fallend, hinsichtlich Cache-Grösse aber monoton steigend verhalten (beginnend beim schnellsten Cache ...). Z.B. bilden 1st-level-Cache --> 2nd-level-Cache --> Controller-Cache eine Cache-Hierarchie.
• Cache-Hit	• Wenn ein Cache die von einem Prozessor benötigten Daten enthält, spricht man von einem Cache-Hit anderenfalls von einem Cache-Miss.
• Cache-Manager	• Ein Programm, das entsprechend Administrationsvorgaben oder policies Caches verwaltet. -----> DAB

• Begriff	• Definition
• Cache-Miss	• --> Cache-Hit
• Cache-Nutzen	• Mit Hilfe eines Caches kann auf Daten, auf die wiederholt zugegriffen wird, ab dem zweiten Zugriff schneller zugegriffen werden. Die Daten werden aus dem Cache geholt und nicht aus dem nachgelagerten langsameren Speicher. Dies setzt voraus, dass eine Instanz existiert, welche u.a. die Zugriffshäufigkeit auf Daten überwacht und bei häufigen Zugriffen diese „frequently used data“ im Cache speichert. --> Cache-Manager ; DAB
• DAB (Disk Access Buffer)	• Ein Programm, das in BS2000/OSD die Caches im Hauptspeicher und im Global Store verwaltet. --> Cache-Manager
• Flüchtiger Cache	• --> Volatile Cache
• ILM	• Information Life Cycle Management: Hinter ILM verbirgt sich die Idee, Daten während ihrer gesamten Verweildauer in einer IT-Infrastruktur entsprechend vorgegebener Usage-Profile (policies) so zu managen, dass der Gesamtaufwand für die Datenhaltung und Datenbereitstellung über die gesamte Verweildauer minimal wird (Total Cost of Ownership). Caching ist offensichtlich ein technischer Teilaspekt der Implementierung von ILM.
• L1-Cache	• Auch 1st-level-Cache genannt. Dies ist in einer Cache-Hierarchie (--> Cache-Hierarchie) der schnellste, teuerste und kleinste Cache. Auf ihn greift ein Prozessor in der Regel synchron zu, d.h. es entsteht bei Hit (--> Cache-Hit) kein process- oder task-interrupt. L1-, L2-Caches sind in der Regel direkt in Hardware realisiert, d.h. sie sind transparent für einen Cache-Manager wie DAB.
• L2-Cache	• Auch 2nd-level-Cache genannt. Ein Cache, der in der Cache-Hierarchie nach dem 1st-level-Cache folgt. Er ist in der Regel langsamer und wesentlich größer. ---> L1-Cache
• LRU-Prinzip	• Da ein Cache nur endlich groß ist, müssen „alte“ Daten, die sich schon im Cache befinden evtl. neuen Daten Platz machen, d.h. diese alten Daten werden auf die nächst niedrigere Stufe der Cache-Hierarchie verdrängt. Das LRU-Prinzip ist die Strategie die am längsten vom Prozessor nicht gebrauchten Daten zu verdrängen (least recently used = LRU).
• Nicht-flüchtiger Cache	• --> Volatile Cache
• Non-Volatile Cache	• --> Volatile Cache
• Volatile Cache	• Volatile Caches (volatile = flüchtig) sind Caches, deren Daten bei Server-HW-Ausfall unter Umständen verloren sind, z.B. bei Stromausfall. L1-,L2-Caches sind in der Regel volatile Caches. Der an BS2000/OSD S-Server optional anschließbare Global Store ist dagegen mit Hilfe von Redundanz als non-volatile-Cache betreibbar.
• Write In-Cache	• In dieser Cache-Betriebsart erfolgt die Ausgabe von Daten zunächst nur an einen Cache und erst später im Rahmen der Verdrängungsmechanismen (--> LRU-Prinzip) auf das Disk-Subsystem. Dies birgt das Risiko, dass bei Cache-Ausfall auf dem Disk-Subsystem die Daten nicht konsistent sind.
• Write-through-Cache	• In dieser Cache-Betriebsart erfolgt die Ausgabe von Daten in einen Cache und gleichzeitig auf das Disk-Subsystem.

Alle Rechte vorbehalten, insbesondere gewerbliche Schutzrechte. Änderung von technischen Daten sowie Lieferbarkeit vorbehalten. Haftung oder Garantie für Vollständigkeit, Aktualität und Richtigkeit der angegebenen Daten und Abbildungen ausgeschlossen. Wiedergegebene Bezeichnungen können Marken und/oder Urheberrechte sein, deren Benutzung durch Dritte für eigene Zwecke die Rechte der Inhaber verletzen kann.
Weitere Einzelheiten unter ts.fujitsu.com/terms_of_use.html

Herausgegeben durch:

Frank Schützhold
Telefon: ++49 (0)89 3222 2627
frank.schuetzhold@ts.fujitsu.com
de.ts.fujitsu.com

Partner login
partners.ts.fujitsu.com

Copyright © Fujitsu Technology Solutions GmbH 2009