T E C H N O T E :
Write Cache Flushing: Techniques for Properly Handling System Shutdown

By Vinnie Moscaritolo

email: vinnie@apple.com
<http://webstuff.apple.com/~vinnie/>

Apple Developer Technical Support (DTS)

Many high-performance disk drives available for Macintosh computers utilize an on-board hardware write cache mechanism. Although these devices can enhance disk performance, they also require special software consideration to avoid the loss of data on shutdown.

This Technote is directed primarily to third-party IDE and SCSI disk interface module software developers, and addresses ways to flush the disk's write cache during the shutdown process.

Write Caching

Many disk drive manufacturers now include a write cache on the drive hardware. This cache is usually an area of temporary volatile storage (RAM) that has a faster access time than the actual drive media. Its purpose is to increase throughput.

These devices use an algorithm to write back the cache. When the write back cache mode is enabled, the disk controller uses the cache to temporarily store data that is promised to be written to the medium at a later time. This allows the disk controller to indicate to the computer that a write command has completed prior to the blocks of data actually being transferred to the medium. The process of transferring the cached data to the drive medium is known as **cache-synchronization**.

Because cache hardware uses volatile RAM, there is a period of time prior to cache-synchronization when the data is vunerable to being lost if a power failure occurs before the media is updated.

Most drives will flush the cache periodically if there is no activity. However, some drives are unpredictable because if you don't explicitly synchronize their cache, they won't do it. This may pose a problem especially at system shutdown. Either the amount of time they wait after idle before synchronizing is sufficiently long enough that the power is shut off, or the Macintosh resets the bus before the data gets written. Delaying shutdown is not a reliable way to flush write cache because of varying vendor implementations.

Macintosh systems that use disk driver software which improperly handles write-caching run the risk of data loss or even corruption of the disks' file structure on shutdown.

The Good, the Bad and the Ugly

Caching is such a popular technique that any particular Mac OS configuration could be used in any number of layers. For instance, the Macintosh File System maintains a cache. Even the disk interface card (e.g., PCI or NuBus) might also have a cache. These can coexist seamlessly, provided that during the shutdown

process they are synchronized in a certain order: *from the inside out.* This is required to ensure data stored actually gets to the media.Yet in the current OS, the driver writer has only limited control of this process. Let's examine some of the available options.

Doing Nothing — Bad

In the past, Apple shipped systems with drives which either did not support write caching or had the cache disabled. However, this is no longer neccessarily true. If your driver still ignores the shutdown synchronization, you will suffer volume corruption.

Disabling the Cache — Ugly

Some disk driver vendors simply attempted to prevent this circumstance by explicitly requesting the drive to select a write-through caching algorithm — in effect, disabling the cache. This method is ugly, since it fails to take full advantage of the drive performance available.

Tail Patching of _UnmountVol — Uglier

Let's assume we enable the cache. The next question is, how do we ensure that our sync code runs after the file system is done with the volume? That's easy, you say, tail patch the _UnmountVol trap and flush the drive after the file system is done with it. In a perfect world, this idea might work. Unfortunately, some developers (including those at Apple) who depend on modifing the behavior of the File Manager had the same idea. Things can get pretty ugly, because there is no guarantee that your cache-synch code will run last.

Shutdown Task — Close, But No Cigar

The Macintosh Process Manager actually has a mechanism that allows code to run before shutdown or reset occurs. By invoking a call to ShutDwnInstall, your driver code can install an entry in the Shutddown Manager's power down queue. The Mac OS will then call your ShutDownProc during the power down process.

I'd like to say it was that simple, but after further examination of the code, I discovered that the File Manager also uses this method. So even though you

are guaranteed to run, you can't be certain if it will be before or after the volumes unmount.

Actually, the solution is close at hand. I mentioned earlier that disabling the cache was one option. But instead of doing it all the time, we just disable caching at shutdown only and then force a synchronization. Even if any transfers happen after our ShutDownProc runs, they won't get caught waiting around in the write cache.

Installing the Shutdown Procedure

To setup your shutdown proc when your driver is opened, instruct the Shutdown Manager to install a task in the `sdRestartOrPower` queue.

```
ShutDwnInstall(YourShutDownProc, sdRestartOrPower);
```

Be sure to load your shutdown procedure into the system heap because the Process Manager frees all applications and other temporary heaps before calling the Shutdown Manager.

Shutdown Procedure for SCSI Devices

Disabling the Drive Cache

The first thing you must do is disable the write cache by resetting the write cache enable (WCE) bit of the caching page, as shown in Table 1.

Table1 Caching page

Bit / Byte	7	6	5	4	3	2	1	0
0	PS	Reserved	Page code (08h)					
1	Page length (0Ah)							
2	Reserved					**WCE**	MF	RCD
3	Demand read retention priority				Write retention priority			
4	(MSB)							
5	Disable pre-fetch transfer length							(LSB)
6	(MSB)							
7	Minimum pre-fetch							(LSB)
8	(MSB)							
9	Maximum pre-fetch							(LSB)
10	(MSB)							
11	Maximum pre-fetch ceiling							(LSB)

You need to use the MODE SELECT command with (SP=0) to ensure that this change does not get written into the disk's non-volatile RAM. You just want the cache disabled for now, as shown in Table 2.

Table 2 Mode select command

MODE SELECT(10) command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (55h)							
1	Logical unit number			PF		Reserved		**SP**
2	Reserved							
3	Reserved							
4	Reserved							
5	Reserved							
6	Reserved							
7	(MSB) Parameter list length							
8	Parameter list length (LSB)							
9	Control							

Synchronizing the Write Cache

You have to force the disk to flush its write cache. Some drivers attempt to take shortcuts here by inserting a delay of a few hundred microseconds, assuming that the drives will flush automatically. Empirical evidence, however, indicates this is not always true. Some drives don't flush unless explicitly told to do so. You can blame this on a loose interpretation of the SCSI spec.

To be safe, send a SynchronizeCache (0x35, 10 byte CDB) command with **all fields 0,** to the drive. This guarantees all of the cache will be consistent with the media. The **Immed** bit must be clear to ensure the command will wait for completion, as shown in Table 3.

Table 3 Synchronize cache command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (35h)							
1	Logical unit number			Reserved			**Immed**	RelAdr
2	(MSB)							
3								
4	Logical block address							
5								(LSB)
6	Reserved							
7	(MSB)							
8	Number of blocks							(LSB)
9	Control							

Because the SynchronizeCache is described in the SCSI specification as an optional command, it's possible but unlikely that a drive could support write caching, but not support the SynchronizeCache command. Developers ought to be aware of this.

Ensuring That All Further Writes Go Directly to the Media

In some drives, disabling the write cache is not preferred. An alternative to doing this is to execute the synchronize cache command and from that point on only issue WRITE commands with the force unit access (FUA) bit set. This ensures that all further writes go directly to the media, as shown in Table 4.

Table 4 Write Command

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation code (2Ah)							
1	Logical unit number			DPO	**FUA**	Reserved	Reserved	RelAdr
2	(MSB)							
3								
4	Logical block address							
5								(LSB)
6	Reserved							
7	(MSB)							
8	Transfer length							(LSB)
9	Control							

The driver must check if the disk supports this feature. You can do this by executing a MODE SENSE command to get the device-specific parameters and examining if the DPOFUA bit is set, as shown in Table 5.

Table 5 Device specific parameter

Bit	7	6	5	4	3	2	1	0
	WP	Reserved		**DPOFUA**	Reserved			

Shutdown Procedure for ATA Devices

Due to loose interpretations and vendor uniqueness in the ATA Standard, there is no defined way that a driver can be assured that the disk's cache has been flushed. And the trigger to force a flush appears to vary by vendor. Even though the Set Features command can be used to disable the write cache, some drives don't flush it until they receive another command.

One way of handling this is to issue a Standby or Sleep command to the drive when you want to flush the cache. This works because the drive must flush the cache before spinning down. These commands may complete before the drive completely spins down, but they do not complete before the cache is flushed.

Unfortunately, at the time of this writing, ATA cache synchronization is still an unresolved issue with the X3T13 Standards Committee.

Summary

Because it is part of the ANSI specification, Macintosh disk driver writers must support write caches and must assume that drives are shipped by manfacturers with write cache enabled.

Write caches may enhance system performance, but it is important to properly handle system shutdown to prevent any loss of data.

Not all drive vendors implement commands the same way. *Therefore, it is the responsibility of the driver writer to understand the specific workings and implementations of the commands unique to a vendor's disk drive.*

Further References

- *Inside Macintosh: Devices,* chapter on "SCSI Manager 4.3"
- *Inside Macintosh: Processes,* chapter on "Shutdown Manager"
- SCSI Spec: ANSI XT3T9.2/375R revision 10L, available at ftp://abekas.com:8080/scsi2/
- ATA-3 Spec: ANSI X3T10/ 2008D Revision 6, available at ftp://fission.dt.wdc.com/pub/standards/ata/ata-3

Acknowledgments

Thanks to Clinton Bauder, Cameron Birse, Jim Luther, and Richard Schnell.