NOTE: This Technical Note has been retired. Please see the Technical Notes page for current documentation.

Technical Note FL32

Hey Buddy, Can You Spare A Block?

CONTENTS

This Technical Note discusses a new feature of the System Software 7.0 Disk Initialization Package--bad block sparing.

Warning:
Software that accesses blocks directly from the disk or makes assumptions about the physical blocks of a device is, has always been, and will always be, a compatibility risk. The format of the file directory is changing in System Software 7.0 and additional changes being made to the Disk Initialization Package will cause such software to fail.

[Feb 01 1991]

Introduction

The Disk Initialization Package that is being shipped with System Software 7.0 contains a new feature, **bad block sparing**. This new feature is done without modification to any disk drivers and is independent of the device's geometry. When the system finds bad blocks, it removes them from the free storage pool so that the file system does not use them. This feature does not affect any applications which use the normal HFS file system; the only expected impact of this feature is for those applications which perform disk reads from or writes to the disk directly, like scavenger, recovery, and floppy disk utilities.

The new feature of the new Disk Initialization Package maps any bad blocks found on any HFS volume. This feature is valuable considering the number of blocks that are on a 800K or 1440K floppy disk. If a single block is bad, the previous Disk Initialization Package would return an error and the system would reject the entire disk. The new Disk Initialization Package may attempt to map any bad block found on a non-Sony drive, but the probability is low that a newly formatted SCSI drive has a bad block. If it were to have bad block remaining after its low-level format, then there's something wrong with the disk and it's most likely a hardware problem. It is possible for a volume to have encountered a bad block during normal use. These can be mapped out by going to the Finder and choosing Erase Disk... from the Special menu, calling `_DIBadMount`, or calling `_DIZero`. Calling `_DIFormat` or `_DIVerify` does not call upon the sparing algorithm.

Note:
When a user chooses Erase Disk... from the Special menu, the Finder calls `_DIBadMount` with the error code in the `evtMessage` set to `noErr`. This causes the sequence of Disk Initialize dialogs to appear. Applications can call `_DIBadMount` to reformat a disk and ensure that bad blocks are spared. This is further documented in the Disk Initialization Package chapter of *Inside Macintosh* , Volume VI.

The sequence of events after calling `_DIBadMount` is as follows. `_DIBadMount` issues a call to the disk driver to perform the low-level format of the disk. This is a `_Control` call with `csCode = formatCC`. Once the driver returns its result, `_DIBadMount` attempts to verify the blocks on the disk. This becomes a `_Control` call to the driver with `csCode = verifyCC`. If the driver returns an error, then `_DIBadMount` begins scanning the disk for bad blocks and mapping them out. Afterwards, a directory is created and the volume information is written to the disk. This completes the process of `_DIBadMount`.

The Disk Initialization Package does not perform the low-level formatting that is required by a SCSI device or any other non-Sony drive. Generally SCSI drivers ignore these `csCode` values mentioned above and return `noErr`. In returning `noErr`, `_DIBadMount` performs no bad block sparing. Thus, it isn't likely that sparing is used by `_DIBadMount` on a non-Sony disk. Such sparing is performed by the utility software supplied with the disk. Issuing the SCSI `Format` command causes the drive controller to perform the low-level format and sparing of any bad blocks in hardware. This is handled by the SCSI formatter software included with the hard disk product. If any bad blocks are suspected on a SCSI device, it is recommended that users format the disk with the supplied SCSI utility. Finally, **good** SCSI drivers map bad blocks dynamically so that any bad block encountered during normal use is removed.

If an application calls `_DIFormat` directly, the Disk Initialization Packages performs no bad block sparing. If the disk does contain bad blocks, the disk cannot be used. `_DIVerify` is used to verify if the disk contains any bad blocks. This disk may be used if these bad blocks are mapped out. To do this, the application has to use either `_DIBadMount` or `_DIZero.`

Back to top

The Algorithm

Disks that are error-free are initialized exactly as before. Only when the driver's verify routine fails during `_DIBadMount` or if `_DIZero` encounters bad blocks is the sparing algorithm invoked. Sparing proceeds by making a second pass over the disk, writing and then reading back a test pattern. Testing is done a single track at a time, as a compromise between speed and wasted space. (Since it is impossible to determine the geometry of a SCSI drive, all disks larger than the Floppy Disk High Density (FDHD) are tested at an assumed track size equal to the FDHD.) If there are any errors or retries during a test, the sectors are deemed bad.

If more than 25% of the disk is found to contain bad blocks, if the I/O errors appear to be due to hardware failure rather than media failure, or if certain critical sectors are bad (see below), then the initialization fails as it would have without sparing. Otherwise, the blank HFS volume structure is written to the disk. Because the sectors touched during this operation are included in the "critical" list, no changes to the code which initializes the logical volume structure are required. After the volume structure has been written, the Disk Initialization Package:

1. Removes the bad spots from the volume bitmap of available free storage.
2. Creates file extent descriptors for the bad spots and inserts them into the volume extent B*-tree so that the free-space scavenging that takes place at volume mount and by Disk First Aid do not attempt to reintroduce the bad spots into the free storage pool. A reserved file ID (5) is used for these extents.
3. Sets a flag (hexadecimal 0200) in the HFS volume header attributes to provide a canonic and simple way for applications to determine whether or not the disk has been spared. The bad extents are described in the B*-tree with reserved file ID=5.
4. For 800K floppies only, the number of allocation blocks is reduced by one (from 1594 to 1593). This change is done to prevent previous Finders from doing disk-to-disk copies physically (sector-by-sector), which would fail trying to copy the bad blocks. The Finder does physical copies only on 1594 block disks as an optimized method of disk copying.

The critical sectors (those that must be good even on a spared disk) include the boot blocks, the HFS master directory and spare master directory, the volume bitmap, and the initial extents for the two HFS B*-trees. In practice, this means that the first 50 sectors and the second to the last sector of a 1440K FDHD disk must be good.

As described later in this Note, the most error-prone region of a floppy disk seems to be the inner tracks on side one (the bottom side), which, unfortunately, is where HFS keeps the alternate copy of the Master Directory Block (MDB). The normal sparing logic rejects an entire track if any sector on it is bad, but to improve the algorithm's effectiveness, the algorithm has a special case for the alternate MDB. Even if there is an error somewhere in the alternate MDB's track, the algorithm does not reject the disk if the sector it is on is useable. This means that with n sectors per track, only about $1/n$ of the disks with damaged inner tracks are rejected by the sparing algorithm. On FDHD disks n =18, so about 95% of the damaged disks can be initialized, assuming only one sector on the track is bad.

Experience Has Shown

In the few weeks after the new Disk Initialization Package was written, Apple had the opportunity to test it against roughly one hundred disks (both 800K and 1440K) that could not be formatted with the standard Disk Initialization Package, or that had developed errors since being formatted. Most of these were successfully spared using the new Disk Initialization Package.

Engineers noticed a trend in the small sample tested: errors on both 800K and 1440K disks are not uniformly distributed across the media. By far the most common place for errors to occur is on side one tracks 75-79, with the second most common place being tracks 0-3. Interior errors seem to be rare. One explanation for the high proportion of errors on side one tracks 75-79 might be that Apple now parks the heads over that area before ejecting a disk, so when a disk is inserted the heads come down there, possibly touching and scraping the media. Side one is probably more vulnerable both because the diameter of a given track is smaller on that side and because the head faces up and therefore easily collects dust.

The Compatibility Risks

Applications that manipulate disks through the HFS documented routines (by far the vast majority of applications) see no difference using spared disks. The only user-visible difference is that the Disk Initialization dialog box shows an additional message ("Re-Verifying disk...") while sparing an imperfect disk, and a spared disk has slightly less free space than an error-free disk. The identified risks associated with sparing are due to the following:

1. Applications that directly manipulate the HFS volume structure (in particular the extent B*-tree, volume bitmap, or the volume attributes field) need to be changed to respect the bad spot extents. In particular, this includes disk utilities such as Disk First Aid, that repair and reorganize HFS volumes. Note that these same applications also need to be revised to correctly handle System Software 7.0 aliases. Apple has a version of Disk First Aid that supports both sparing and aliases, and it ships as part of System Software 7.0.
2. Applications that physically access disks, sector by sector, do not work if the disk contains a sector that has been spared. The best, and possibly only, examples of this class of application are the disk duplicating utilities.

Back to top

Summary

Unfortunately, the first and last tracks (the ones most likely to be bad) are also the very tracks that are critical to the HFS layout and which, therefore, must be error-free. As a result, the sparing algorithm is not as effective as it might be. To minimize this problem in the future, Apple intends to change the parking space to which the Sony drivers move the heads before ejecting a disk, this time from cylinder 79 to cylinder 40. Cylinder 40 is probably a slightly better choice, as the media is more flexible at this point since it is farther from the rigid metal hub, yet it is far enough away from the outer edge to avoid the problems experiences with cylinder 0. More importantly, though, cylinder 40 does not cover any sectors that are critical to HFS.

The decision not to create a directory entry for the bad spot file has both advantages and disadvantages. The major advantage is that logical operations such as directory enumerations and file-by-file disk copies are completely unaware of the bad spots. Since Finder disk copies are file-by-file, this is important. The major disadvantage is that Disk First Aid and other third-party disk utilities do need to be upgraded to recognize and avoid the bad spots, even though they are not part of a file. The required changes, which are very simple, have been incorporated into the System Software 7.0 version of Disk First Aid.

Another, completely independent way to deal with imperfect media is at the driver and controller level. If the driver reserved a few cylinders for re-vectoring bad tracks, it could present the appearance of error-free media to its clients, including HFS and the Disk Initialization Package. Most, if not all, hard disk drives already do this. This is a very attractive solution, because no changes to higher-level software are required since the traditional Macintosh model of error-free media is preserved. However, it is a much more difficult task to modify the several existing floppy disk drivers (such as the 6502-based IIfx driver) than it was to enhance the Disk Initialization Package. Also, there are major

compatibility problems involved with driver-level solutions since old drivers would not be able to correctly handle revectored tracks. Fortunately, this is not an mutually-exclusive decision: the disk initialization algorithm described here does not prevent later improvements to the drivers.

It should be noted that sparing in Disk Initialization Package does not address problems that occur after a disk has been formatted. This class of dynamic errors is much harder to deal with, since neither the Macintosh OS file system nor its clients are set up to handle such I/O errors gracefully, if at all. On the other hand, many operating systems apparently spare only during initial formatting. Media errors encountered during an I/O operation are handled by the driver code.

Back to top

References

Inside Macintosh , Volume II, Disk Initialization Package

MPW SonyEqu.a interface files

Back to top

Downloadables

 Acrobat version of this Note (1K) Download

Back to top
