swissbit®

Flash SSDs Mastering

Application Note

BU:	Flash Products
Date:	April 20, 2018
Revision:	1.2
File:	Swissbit AppNote Flash SSD Mastering-Rev1.2.docx

Content

1 Overview

This document provides a guideline to optimal mastering of a Swissbit flash based SSD, duplicating the content of a master drive.

Mastering (duplicating the contents of a Master Drive) of an SSD is not as straightforward as it is when mastering a rotating hard disk drive. This document presents some technical background to understand the problems that might arise out of this process (see chapter 2) as well as a possible recommended process for mastering drives (see chapter 3).

2 Background

2.1 Flash Memory Organization

When looking at how a Flash memory is addressed, the smallest logical/administrative unit is a sector. A host addresses a flash memory drive by using sector addresses. For the flash, however, the smallest unit is a page. A typical page size of flash devices is 16Kbyte. The next level of hierarchy are blocks which include several pages. A block is the smallest unit of memory that can be erased. A typical block size of current flash devices is 4MBytes or 256 times 16Kbytes pages.

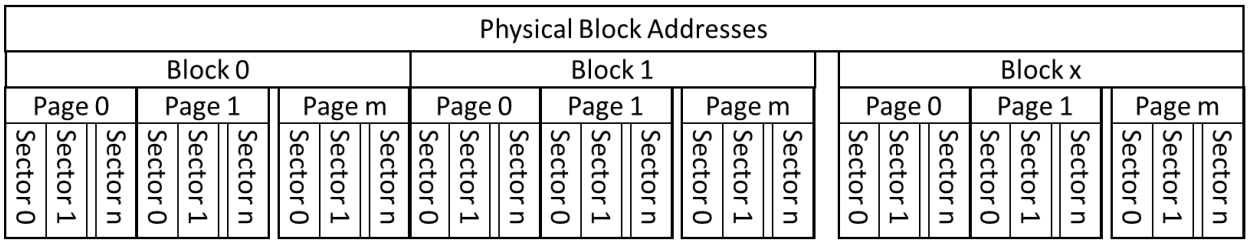

Figure 1: Organization of a Flash Memory

While Flash memory can be written / programmed at page level, it must be erased at block level. Erasing generally sets all bits in the block to 1. Starting with a freshly erased block, the block must be programmed incrementally by pages, filling it up with data. However, once a bit has been set to zero, it can only be changed back to one by erasing the entire block. A page can be programmed only a single time after having been previously erased.

As blocks are the "management" or "administrative" units, blocks will wear out as memory cells degrade after a number of erase cycles. When defective, these blocks will be considered "bad blocks".

2.2 Flash Memory Controller and Firmware

The complex nature of Flash cells and their organization demands reliable, high performance control functionality. Flash Controllers of all kinds consist of an interface to the Flash memory, a processor and a host interface.

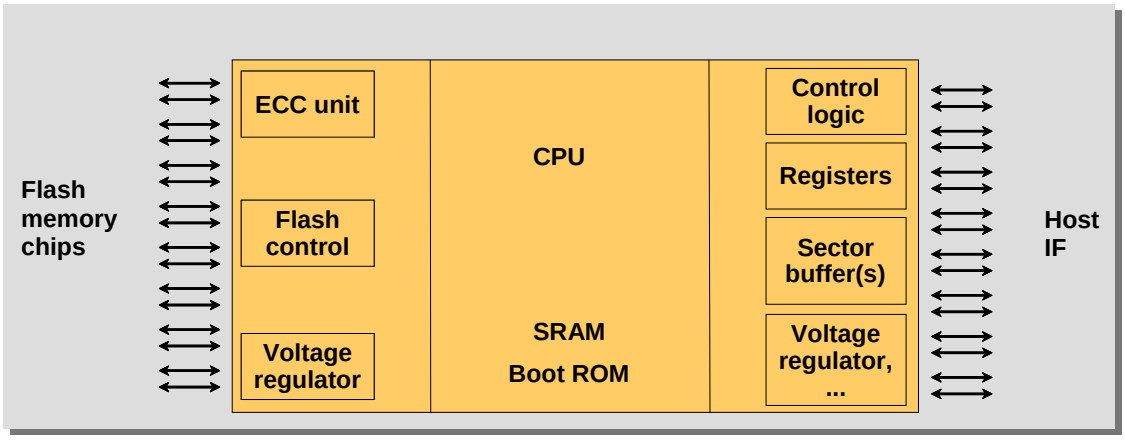

Figure 2: Generic Block Diagram Solid State Memory Solution

The controllers are based on a CPU together with dedicated hardware blocks, including an error correcting code (ECC) unit, buffers, Flash and host interface control logic.

2.3 Overprovisioning

Overprovisioning in a flash memory drive is the difference between the physical size of the internal flash memory and the logical size of the memory that is made available to the host. All flash memory drives have some overprovisioning – the internal flash memory is actually bigger than the memory available for the host. The part of memory that is not available to the host is used for internal operation such as garbage collection and the storage of all internal management structures. Typically SSDs have between 7 and 15% overprovisioning.

2.4 Data management and garbage collection

In order to write new data to the flash, the flash memory controller firmware will have to allocate an erased flash memory block where the data can be written to. As data comes from the host, it is written sequentially into this erased memory block. In parallel the management structures are being updated to keep track of the physical location of the data written. To keep things simple, in this document we will not consider these management accesses, but we will consider only the user data.

If the host continues to write to the drive, at a given point all user available space will be occupied and the internal firmware can only work with the overprovisioned free area. This state is called saturation.

If the host writes new data in this state, it will certainly overwrite some existing data (since the drive is fully written). In this situation the flash memory controller firmware will allocate an erased block of the overprovisioning area and write the data to it. At this point the previously assigned page for the just written address can be discarded. Since a flash can only be erased in blocks, this cannot be done immediately. The controller firmware will mark the data as "old" and continue its work. The page containing old data that can be discarded is typically called a dirty page.

If the host continues writing new data, the drive will reach a situation where also the overprovisioning area is nearly completely occupied and the controller firmware has to allocate a new flash block. This is done by selecting a block with as many as possible dirty pages, copying the non-dirty pages to a new block and erasing the original block. Now the original block can be used to write new data. This process is called garbage collection. Note that the efficiency of garbage collection depends on the number of dirty pages found in the collected block, and this will depend on the size of the overprovisioning and on the way the host overwrites the data. In a best case situation a block can be found which is completely dirty so nothing has to be copied.

2.5 Write amplification factor

The write amplification factor is the factor between the amount of data written from the host and the amount of data internally written in flash. If a write operation can be executed immediately since there is an erased flash block that can be used, this factor will be close to one. However, if the write operations triggers a garbage collection operation that requires moving several megabytes of data internally before the host data can be written, the write amplification factor for this operation might be very high.

2.6 TRIM

Erasing a file on a filesystem has been done traditionally by just erasing the entry of this file in the filesystem entry table. Since rotating hard drives do not have to erase the data before writing to it, this strategy is appropriate for them. However, for flash drives this causes a problem.

Since the flash memory controller does not know about the file system (the controller handles only with addresses containing some data, but it does not interpret this data) the area where the erased file was written will still contain valid data in the eyes of the flash memory controller. In this case, after a file has been erased, the Flash data of this file will not be considered dirty. It will be treated as valid data and copied when the block will be emptied by the garbage collection, reducing significantly the efficiency of the garbage collection algorithm, increasing the write amplification factor and ultimately reducing the drive performance and expected lifetime.

To solve this problem modern OSs and drive interfaces have some kind of command to discard old data. In the case of SATA drives the command is called TRIM and it is supported by almost all state of the art OSs.

There are different Trim strategies in the different OS. Some OSs trim the area left by an erased file immediately after it has been erased, and some others have a periodic daily or weekly task that trims all the free areas in the drive.

Having a properly configured OS that correctly trims the free area of a flash memory drive is a key factor to improve the efficiency of the garbage collection, and thus the performance and expected lifetime of a flash drive.

3 Drive mastering

A common setup in a production environment to produce multiple drives with a predefined data set (software and/or data) is to have a master drive with the desired data and duplicate this drive to all the other drives/systems.

There are two types of tools/setups to accomplish this kind of drive cloning: intelligent tools and dummy duplicators.

An intelligent duplicator can typically copy a source drive to a smaller target drive, or even to a drive using a different filesystem. This kind of tools do understand the filesystem of the source and target drives, read only the data that has to be copied and write it to the target drive. This kind of tools do not present any problem since they only copy relevant sectors.

On the other hand, there are dummy duplicators that typically are only able to duplicate a drive if the target is exactly of same or bigger size than the source drive. Unlike the intelligent duplicators, this kind of tools does not understand the filesystem of the target or source drive. It will read every single sector of the source drive and write it at the target drive(s) to the exact same location, no matter what this data is, or even if there is no valid data at this address.

After this kind of duplication, the logical data on both drives will look exactly the same, however the internal state of the drives might vary significantly. Independent of the amount of trimmed or unused space in the source drive, after the duplication the target drive will be always in saturated state, since every single address of the user area has been written by the duplicator.

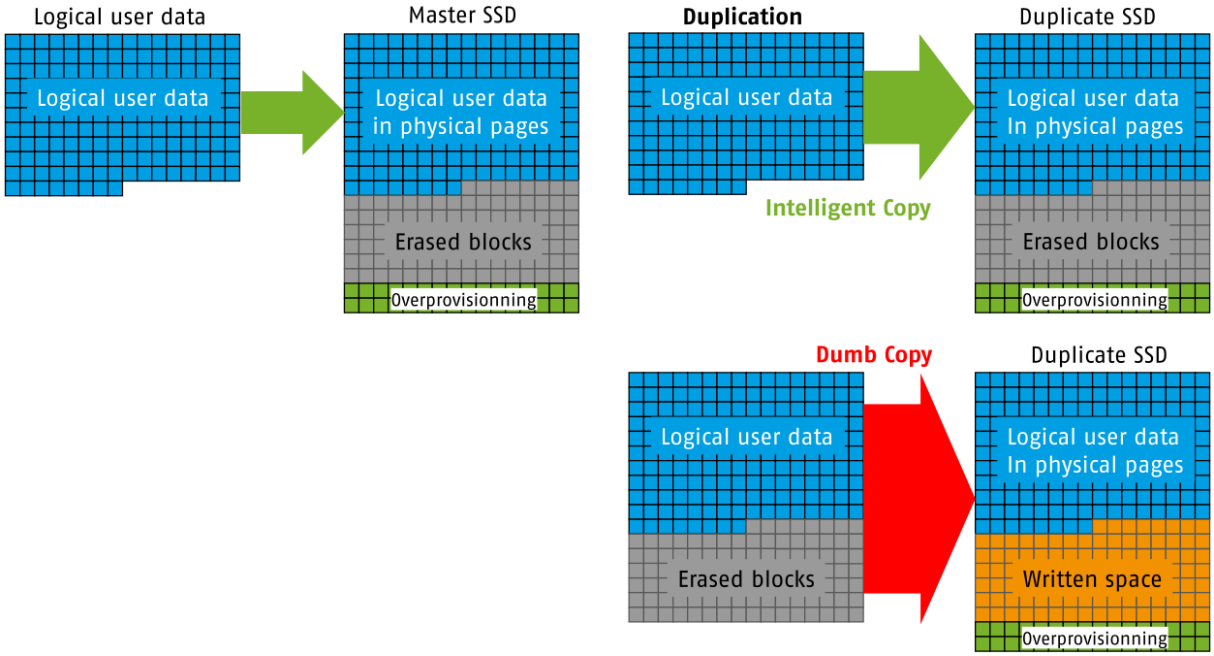

Figure 3: Intelligent vs. Dump duplication

As previously explained, this will lead to a very inefficient garbage collection, high write amplification, very low performance, and significantly reduce the expected lifetime of the target drive.

Even if Trim is supported by the target system used after duplication, the efficiency of the garbage collection will be severely impacted if the drive is in this status. The standard trim strategy of current OS is to trim the space left by an erased file, and this will not recover the drive to a clean state unless the application writes and erases exceptionally big files.

An additional step is needed after the drive duplication in order to release the free area of the drive. The process is depending on the OS of the machine where this step is done.

For Windows 8, 10 or Server 2012 and newer, this can achieved by executing:

```
defrag DriveLetter: /L
```

in a console box with administrative privileges.

Under Linux:

```
fstrim /dev/sdXX
```

on a mounted device will accomplish the same result.

To execute this step, the drive must be connected to a native SATA port of the host system. Using a USB adapter will not work. Please note that the machine (and its OS) where this step is executed does not need to be the same as the target system for the drive.

4 Document History

Table 1: Document Revision History

Date	Revision	Details
13-Nov-2017	1.0	First release
15-Nov-2017	1.1	Chapter 3 – enhanced OS specific info
20-Apr-2018	1.2	Formal changes