NAND FLASH SSD RELIABILITY VERIFICATION

Mihir Shetye, Umang Shah, Prof. Morris Jones

Department of Electrical Engineering, San Jose State University, San Jose, California 95192.

Introduction

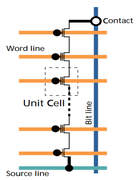
SSD or Solid State Drives came into existence in early 1970s and was used either in military applications or used in research labs for advanced computer systems. In the last ten years the SSD market has grown to more than 15 billion dollars maintaining a double-digit growth pattern every year. The conventional Hard Disk Drives have become almost obsolete and all the new technology is considering SSDs as the default storage. Both, HDDs and SSDs have their pros and cons. The SSDs market is not limited to single PC, laptop or tablet market.
Shorter life is one major drawback pertaining to SSD NAND FLASH memories. This paper proposes an improvised wear leveling algorithm that aims to equate write-erase cycles among memory blocks thereby improving overall reliability of SSDs. The developed algorithm is tested against data obtained using *blktrace* by rebuilding a Linux kernel so as to ensure proper operation.

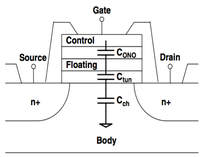
Implement a wear leveling algorithm which would evenly distribute write-erase cycles among all blocks of a SSD memory module. Test the developed algorithm thoroughly against various real life situations. Comparing best and worst case scenarios wherein the SSD blocks corrupt one after the other, the project could be considered a success a value lying between the mean and best case situation is obtained. Validation of this hypothesis could prove to be pretty significant in the current market considering the boom of SSDs and their underlying issues.

Methodology

NAND Flash Architecture Overview

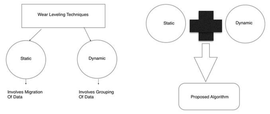
In NOR Flash, the cells are arranged in a parallel manner. It is due to this difference between NAND and NOR Flash, NAND Flash memories are usually around 60% more compact or smaller than NOR Flash. The grouping of cells in NAND Flash is usually in multiples of 16 or 32. There are two special transistors placed at the ends of a line to make sure the line is connected or tied to ground and bit-line respectively.





Methodology

Proposed Algorithm



Most of the SSDs today, have only one kind of wear leveling implemented, which is either static or dynamic. As said above static algorithm involves migrating data and dynamic wear leveling involves directing the data to the best option of destination block available dynamically. Our algorithm is a combination of both, which involves integration of positive aspects of static as well as dynamic wear leveling techniques. Our algorithm categorizes data dynamically, and it also activates the static wear leveling techniques when blocks cross certain threshold conditions. It maintains pools of blocks depending on the erase counts of each of them. These pools then lay certain constraints for static wear leveling. Also most of wear leveling algorithms implement their process flow over number of program operations whereas this algorithm has been implemented on the number of erase counts for each block.

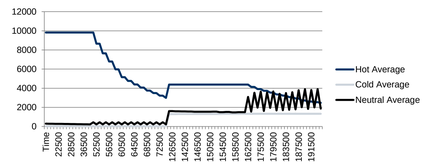
State Diagram

Condition 1: erase_count[neutral_var1] > neutral_avg + Threshold_Grouping
Condition 2: erase_count[neutral_var2] < neutral_avg – Threshold_Grouping
Condition 3: erase_count[hot_var1] > hot_avg + Threshold_Migration
Condition 4: erase_count[cold_var1]

Grouping is performed dynamically and controlled by conditions 1 and 2.

Results

Results obtained are depicted in graphical form. It can be see that as the number of erase cycles increase the average erase counts of cold, hot and neutral pools are plotted. In the figure shown below we can clearly see that when number of write operations on hot blocks increases the average erase counts of hot pool is decreasing. This clearly shows that the wear-leveling algorithm is migrating data and other blocks lying in the neutral pool that are cold or inactive are being used up. Hence, the wearing is equalized and the hot blocks are prevented from further being damaged or worn out. This kind of behavior clearly ensures higher reliability in memories.



The plot above also shows how the cold averages vary with increasing erase counts, Post block grouping the average erase counts of cold blocks increase and while they are being migrated the average of erase counts is increasing with a small slope. This is because the number of blocks in cold pool is high and the increase of erase counts is also a small value and hence the overall increase in average value becomes small. But since there is an increase it clearly shows that cold blocks are being migrated and swapped with hot blocks and hence they are trying to distribute the damage that has been caused to the hot blocks.

The average erase counts of blocks in the neutral pool can also be seen above. This curve clearly shows that when hot blocks from the neutral block are migrated to the coldest block in the cold pool and when coldest block from the neutral pool is being swapped with the hottest block in the hot pool the average erase counts of neutral pool increase and decrease respectively. Block regrouping in neutral pool is depicted by the rapid rise in slope in the middle of the curve when new random data is fed by the test bench. Past this point, the graph curve shows a choppy behavior like before thereby explicitly implying that the algorithm is migrating data and trying reduce the wearing problems.

Summary

The aim of this project was to implement a wear-leveling algorithm that involved both static and dynamic wear leveling techniques and such an algorithm was implemented. With the help of the results shown above it is clear that the algorithm is trying to distribute the wearing of blocks evenly across the whole memory by making sure that the erase counts of the blocks are not too high or too low. Also grouping of blocks and data migration or block swapping is being performed. The algorithm was built using System Verilog, the test bench was also written in System Verilog and two Perl scripts were written to obtained test data and generate random test data for the module.
In conclusion the results obtained prove that the average erase counts of hot blocks is reducing as they are being swapped with the cold ones when they are subjected to lots of erasure over a period of time. This is sufficient enough to prove that the overall reliability of SSD NAND Flash would increase when the algorithm used rather than no wear leveling being implemented.

Key References

[1]. Reliably Erasing Data From Flash-Based Solid State Drives by Michael Wei, Laura M. Grupp, Frederick E. Spad, Steven Swanson

[2]. GANGRENE: Exploring the Mortality of Flash Memory by Robert Templeman and Apu Kapadia

[3]. Rejuvenator:A StaticWear Leveling Algorithm for Flash memory by Muthukumar Murugan and David Du

[4] Write Amplification Analysis in Flash-Based Solid State Drives by Xiao-Yu Hu, Evangelos Eleftheriou, Robert Haas, Ilias Iliadis, Roman Pletka

[5] Nonvolatile Memories: NOR vs. NAND Architectures by L. Crippa, R. Micheloni, I. Motta and M. Sangalli

[6] Introduction to Flash Memory by Roberto Bez, Emilio Camerleghi.

Acknowledgements

The authors wish to thank Professor Morris Jones for his guidance and assistance in this project.