Flash-memories in Space Applications: Trends and Challenges

Politecnico di Torino

Flash-memories in Space Applications: Trends and Challenges

Authors: Caramia M., Di Carlo S., Fabiano M., Prinetto P.,

Published in the Proceedings of the EEE 7th East-West Design & Test Symposium (EWDTS), 18-21 Sep. 2009, Moscow, RU.

N.B. This is a copy of the ACCEPTED version of the manuscript.

Flash-memories in Space Applications: Trends and Challenges

Maurizio CARAMIA(*) Stefano DI CARLO (+) Michele FABIANO(+) Paolo PRINETTO(+)

(*)
Thales Alenia Space
Command Control and Data Handling
Torino, Italy
Maurizio.Caramia@thalesaleniaspace.com

(+)
Politecnico di Torino
Dipartimento di Automatica e Informatica
Torino, Italy
{Stefano.Dicarlo, Michele.Fabiano, Paolo.Prinetto}@polito.it

***Abstract* – Nowadays space applications are provided with a processing power absolutely overcoming the one available just a few years ago. Typical mission-critical space system applications include also the issue of solid-state recorder(s). Flash-memories are nonvolatile, shock-resistant and power-economic, but in turn have different drawbacks. A solid-state recorder for space applications should satisfy many different constraints especially because of the issues related to radiations: proper countermeasures are needed, together with EDAC and testing techniques in order to improve the dependability of the whole system. Different and quite often contrasting dimensions need to be explored during the design of a flash-memory based solid-state recorder. In particular, we shall explore the most important flash-memory design dimensions and trade-offs to tackle during the design of flash-based hard disks for space applications.**

I. Introduction

Nowadays space applications are provided with a processing power absolutely overcoming the one available just a few years ago. However the very strict requirements have often driven the design choices toward older and/or lower-performing radiation-tolerant electronics. Although each new space application has its own story and increasing requirements [1], a typical mission-critical space system application includes, among the several aspects, the design of a solid state recorder(s): this issue is addressed in this paper.

Flash-memory based systems are gaining acceptance and usage not only in the consumer market but in space applications, as well, where they could play the role of high-capacity storage devices: in fact flash-memory guarantees both the non-volatility in case of power loss and a highest storage density, being at the same time shock-resistant and power-economic [2].

However designing flash-based systems for space application requires both exploring a huge number of design dimensions and evaluating a huge amount of trade-offs among all such dimensions. The most relevant dimensions include flash-memory technology, flash-memory architecture, file management system, dependability enhancement strategies, power consumption, weight, physical size. This paper aims to explore the most peculiar flash-memory design dimensions and trade-offs to tackle during the design of flash-based hard disks for space applications.

The paper is organized as follows: Section 2 addresses the main flash-memory peculiarities, while Section 3 explores the dimensions of the issues of designing flash-based mass memory devices, focusing also on the space applications.

II. Flash-memory Characteristics

Flash-memories present several interesting features that properly fit with the requirements of mass-memories for space applications; possible alternatives also need to be evaluated.

Our solid-state recorder would need a relative high capacity: a first possible solution could be DRAMs. On the one hand DRAMs are very fast, reliable and provide a very high data rate, but on the other hand they need a battery pack-up to not lose data and this issue generate an intricate balance between battery mass and data retention time: data retention over years is not feasible and count of battery charge cycles are limited. DRAMs are not discussed anymore in this paper.

The second and more attractive solution is the use of flash-memories. There are two major types of flash-memory in the current market: NOR and NAND flash-memory. NOR flash-memory is for EEPROM replacement and is more suitable for program execution, while NAND flash-memory is more suitable for storage systems [8], [5]: Table 1 briefly sums up the main characteristic of these types of flash-memory.

This paper addresses only NAND flash-memories: in fact they are the most suitable choice for HD replacement.

On the one hand, flash-memories are non-volatile, shock-resistant, and power-economic: a pure flash-memory based solution could guarantee unlimited data retention time and no need of battery backup. On the other hand flash-memories are still much more expensive than hard disk (HD), only rather large data structures could be accessed and, in addition, DDR2 SDRAMs provide higher write/read rate (e.g., 6 Gbit/s) compared to the moderate one accomplished by flash-memories (e.g., up to 80/200 Mbit/s) [2]. Moreover one of the main challenging aspects of flash-memories is that the space already written (i.e., programmed) with data usually cannot be overwritten unless it is erased from the flash-memory device.

A NAND flash-memory is usually partitioned into blocks: each block has a fixed number of pages and each page has a fixed size. A block is the smallest unit for erase operations, while read and write operations are done in terms of pages, i.e.,

a page can be erased only if its whole corresponding block is erased, whereas a page can be read/written independently.

In addition flash-memory wears out after a certain number of erasure cycles (i.e., actually 10^6 for NAND flash-memory): if the erasure cycles of a block exceed this number, it becomes a "bad block" and is not reliable for storing data anymore.

Finally another peculiar aspect of flash-memories is that technology provides the possibility of storing more than one bit of information per cell: in fact traditional flash-memory devices (Single-Level Cell or SLC) store only one bit per cell, while newer devices (Multi-Level Cell or MLC) are able to store typically two bits per cell.

III. Issues in Designing Flash-based Hard-Disk for Space Applications

When a flash-based system for space application has to be designed, the investigation of a vast quantity of design parameters needs to be defined. A possible partial taxonomy of such parameters is discussed in the sequel of this section, focusing in particular on:

- Flash-memory Technology
- Flash-memory Architecture
- Flash-memory Wearing, Testing and Dependability
- Using flash-memory as Hard-Disk
- Hard-Disks in Space Applications

A. Flash Technology

First of all designers should choose between NOR and NAND flash-memory. Since the issue of solid-state recorder(s) for space applications is addressed, the most suitable choice is adopting NAND technology: the main reasons are at the same time the non-volatility and the highest storage density [2], with a peculiar shock-resistance and low power-consumption.

B. Flash Architecture

Designers have to decide the size of the *blocks* and the number of the *pages* for each block. This may imply selecting the most appropriate flash-memory chipset.

C. Flash-memory Wearing, Testing and Dependability

There are some common issues to address during the design of a flash-based mass memory device, like *wearing, testing and dependability*: they are strictly linked among each other and designers should evaluate the right trade-off among them.

Wear Leveling – Several approaches have been proposed to tackle the problem of "wearing". Among all, a significant role is played by the so called *wear leveling* techniques: it is aiming at distributing data evenly across each memory block of the entire flash-memory to avoid single block to wear out.

Wear leveling techniques like [15] – [22] need to be considered or higher capacity flash-memory devices could be used, then especially taking care of the resulting drawbacks in terms of weight and volume [2]. A comparative analysis of some wear leveling algorithms could be found in [18].

	NAND	NOR
Standby Power	Low/Med	Low
Active Power	Low	Med/High
Cost per bit	Low	High
Read Speed	Med/High	High
Write Speed	High	Low
Erase Speed	High	Medium
Capacity	High	Low
Erase Cycles	10^6	10^5
File Storage Use	Easy	Hard
Code Execution	Hard	Easy
Interface	I/O-like	SRAM-like

TABLE 1 – NAND VS NOR FLASH-MEMORY

Testing flash-memories – Flash-memory testing is quite different from testing other kinds of memory.

During read/write/erase operations, flash memories can experience disturbances or faults that do not conform to any of the traditionally known fault models used in testing RAMs. Disturbances along the word-line (WL) and the bit-line (BL) are more critical in flash memories as compared with RAMs, because during program (i.e., writing 0) and erase (i.e., writing 1) operations, high voltages are applied to them. [37]

Specific fault models are needed to properly represent the most frequent physical defects. The flash memory specific faults are derived from [36] and the most significant ones include:

- *program disturb faults*, i.e., disturbance faults that can occur during the programming (writing 0) of a single cell; they are called word-line program disturbance (WPD) and bit-line program disturbance (BPD) [31];
- *erase disturb faults*, i.e., disturbance faults on erase operations; they are called word-line erase disturbance (WED) and bit-line erase disturbance (BED) [31];
- *read disturb faults*, i.e., disturbance faults occurring on read operations; it is referred as RD fault;

Both *program* and *erase disturb faults* are occurring in a cell sharing a common word-line (a *row*) or bit-line (a *column*) with the programmed/erased cell: all these disturbances are able to modify the original value stored inside a cell into another one. The Read-Disturbance (RD) fault occurs on the selected cell when its state changes after consecutive reads. Moreover an Over-Erase Disturbance (OED) occurs when a cell is overly erased such that its threshold voltage is low enough to turn the cell into a depletion-mode transistor [36].

Some conventional RAM fault models may also be used for flash memories, such as stuck-at fault (SAF), transition fault (TF), stuck-open fault (SOF), address decoder fault (AF), and state-coupling fault (CFst) [38].

Secondly efficient test algorithms are needed. Several approaches were proposed for testing NOR flash-memory: in one of the first works [26], the authors proposed *Exclusive Faults* (EF) and *General Faults* (GF) algorithms to detect disturbance faults. Later, in [27] flash-memory disturbances

were modeled as special types of coupling fault: A March-like flash-memory test, called *Flash-March* and its improvement *March-FT* were proposed to detect all fault types in flash-memory. Finally, in [28], an improved test/diagnostic algorithm, called *Diagonal-FT/Diagonal-FD* was proposed to diagnose and distinguish among the disturbance faults, with the help of RAMSES simulator [29]. EF, GF, Flash-March, March-FT and Diagonal-FT algorithms are able to detect both specific flash-memory faults and the same faults as in traditional RAMs, i.e., SAF, TF, SOF, AF and CFst [28].

Finally Built-In Self Test (BIST) and Built-In Self Diagnosis (BISD) circuits have to been taken in consideration: quite a lot of strategies and approaches were adopted for testing NOR flash-memory [38] – [40].

D. Using flash-memory as Hard-Disk

Several challenging aspects need to be addressed when using a flash-memory as a mass-memory device: a possible taxonomy is addressed in the sequel of this paragraph.

Operating System Management – Typical system architecture of flash-memory-based file systems has been proposed [22]. Operating systems (OS) and applications were developed in order to operate with magnetic hard-disks.

With NAND flash-memories, a new class of mass-memory devices was born: on the one hand they are physically totally different from magnetic hard-disks, but on the other hand they both have been thought to be a mass-memory device.

OS and applications were and are able to communicate with common hard-disks. Proper solutions are needed in order to let OS successfully communicate with NAND flash-memory devices: *block-device emulation* and the development of a *native flash file system* are the two alternatives [8].

Block-device emulation approach lets OS to work with a flash-memory device in a HD-like way: it is adopted to accomplish compatibility among the most typically used OS and the flash device, which is seen as a contiguous array of storage blocks. This is an illusion provided by the so called Flash Translation Layer (FTL). Different and peculiar implementations of FTL were addressed [4] – [7], [22].

If dependability is more important than compatibility, then it makes sense to design an entire file system, a *native flash file system*. It does not need an explicit type of FTL, does not work trough a block driver layer and is flash-friendly [21].

Conventional file systems (e.g., FAT) have been designed because HD tend to seek quite slowly. The so-called log-structured file systems [14] have been designed also to reduce seek times managing its storage like a circular log. However flash-memories do not have the seek-time issue and derisive gain are possible: some optimizations could be done according to how flash-memory works, improving performance.

JFFS, JFFS2 and YAFFS implement native log-structured file systems. JFFS and JFFS2 [13], [20] are used for flash-memories in embedded systems: they work well on small devices, but slow down on larger ones. Finally YAFFS(2) [21] is the first file system specifically designed for NAND flash-memory, focusing on data integrity and high performance. Comparisons between JFFS2 and YAFFS were proposed [21].

Few file-systems are qualified for space applications [2]. They are designed to get a high level of data consistency: they are usually ad-hoc for the specific interplanetary mission.

Address Translation – Flash-memories contain data which are usually referred with the help of both logical and physical addresses. Data should always have the same logical addresses, even if they are updating: only physical addresses should be modified. However this translation process has to be efficiently implemented for fast operations: this issue is called *address translation*. Implementations, both in FTL and in native flash file system, need to provide an efficient service.

Bad block management – When a block exceeds the maximum number of erase cycles, it is marked as a *bad block*. In addition vendors supply NAND flash-memories with some bad blocks, because this obviously leads to a significant reduction in yield costs. However in both cases bad block management has to be addressed: bad blocks have to be detected and excluded from active memory space [30]. Simple techniques to handle bad blocks are commonly used [30], [35].

Garbage Collection – The need of erasing data to modify them leads to many challenging issues. At a certain point free-space is going to run out: invalidated pages have to be erased in order to free some space and the only way to erase them is to erase the whole block they belong to. As a consequence valid pages of the block need to be kept safe somehow and somewhere, then the block can be erased and a new free block is finally ready for being used for the requested operations. This issue is referred as *garbage collection*.

Reclaiming invalidated space is a critical aspect of flash-memory design. Flexible cleaning algorithms [32], greedy policies [33], aging functions [19] or periodical collection approaches [34] can be adopted to minimize the cleaning cost.

E. Hard-Disks in Space Applications

A solid state recorder for critical space missions needs to satisfy many different constraints, including, among the others, no loss of mass memory data and the guaranteed availability of storage capability at End-Of-Life (EOL). A well-designed flash-based memory system can meet the requirements of interplanetary missions, but its design must compensate for flash's shortcomings in speed, radiation tolerance, noise, and read/write cycle life and this compensation leverage the costs.

On the one hand there is the need of flash-memories physically qualified to survive in the space environment: vendors should absolutely provide them, with the help of proper strategies and measurements [9] – [12].

On the other hand data integrity, reliability, simplicity, modularity, and autonomy are just some of the key features to fulfill: it is absolutely fundamental also to provide reliable storage of context data, also during spacecraft power outage and in case of a faulty spacecraft computer. Moreover the application of the solid-state recorder should cover a range going from "stand-alone" memory to "embedded" memory. Typical system demands of a space-critical mission have various aspects to be addressed: different data types need to be

considered, mass and power constraints exist, performances do not have to go under a certain critical limit and direct access from ground and On-Board-Computer (OBC) usually need to be guaranteed, as well as indirect access from users.

EDAC - Error Detection And Correction (EDAC) is respectively the ability to detect the presence of errors and to correct them. Designers should evaluate the most proper choice for their design, addressing many issues: the most significant ones include evaluating the type of code to adopt, choosing the number of bits needed for that code (i.e., for accomplishing the requested level of dependability) and addressing where that code has to be stored.

Several ECC algorithms for error checking and correction of NAND flash were proposed, based on Hamming codes or on Reed-Solomon codes [23] – [25].

IV. Conclusions and Future Works

The most significant trends and challenges of using flash-memories in space applications have been addressed in this paper: these concepts are fundamental in order to develop a powerful design environment aimed at supporting the design of a flash-based mass-memory device for space applications.

V. References

[1] Anthony Lai: "Space-ready, radiation-tolerant processor modules: A COTS technology strategy", *Military Embedded Systems Resource Guide, May 2005*

[2] Cassel M., Walter D., Schmidt H., Gliem F., Michalik H., Stähle M., Vögele K., Roos P. Casel.: "NAND-Flash-memory Technology in Mass Memory Systems for Space Applications", *Proceedings Data Systems In Aerospace (DASIA) 2008, Palma de Mallorca, Spain, 2008*

[3] Wu M., Zwaenepoel W.: "eNVy: a NonVolatile main memory storage system" *Proc. Fourth Workshop on Workstation Operating Systems, 1993, 116-118*

[4] Intel Corporation, Technical Report: "Understanding the Flash Translation Layer (FTL) Specification", *December 1998*

[5] Hsieh Jen-Wei, Tsai Yi-Lin, Kuo Tei-Wei, Lee Tzao-Lin: "Configurable Flash-Memory Management: Performance versus Overheads" *IEEE Transactions on Computers, Vol. 57, no. 11, 2008*

[6] M-Systems: "Flash-memory Translation Layer for NAND Flash (NFTL)", *1998*

[7] Intel Corporation, Flash File System, US Patent 540, 448

[8] Chang L. P., Kuo T. W.: "An efficient management scheme for large-scale flash-memory storage systems", *Proceedings of the 2004 ACM Symposium on Applied Computing , Nicosia, Cyprus, 862-868, 2004*

[9] Brüggemann T., Schmidt H., Walter D., Gliem F., Michalik H.: "Further Heavy Ion and Proton SEE Evaluation of High Capacity NAND-Flash-memory Devices for Safeguard Data Recorder", *8th ESA/ESTEC D/TEC-QCA Final Presentation Day, February 2007*

[10] Schmidt H., Walter D., Brüggemann M., Gliem F., Harboe-Sørensen R., Virtanen A.: "Heavy Ion SEE Studies on 4-Gbit NAND-Flash-memories", *Radiation Effects on Components and Systems (RADECS) 2007, DWL-14, September 2007*

[11] Schmidt H., Walter D., Gliem F., Nickson B., Harboe-Sorensen R., Virtanen A.:"TID and SEE Tests of an Advanced 8 Gbit NAND-Flash-memory", *Proc. IEEE Radiation Effects Data Workshop, 2008, 38-41*

[12] Brüggemann M., Schmidt H., Walter D., Gliem F., Harboe-Sørensen R., Roos P., Stähle M.: "SEE Tests of NAND Flash-memory Devices for Use in a Safeguard Data Recorder", *Radiation Effects on Components and Systems (RADECS) 2006, A-3, Volume A-3, 2006*

[13] Woodhouse D., Red Hat, Inc.: "JFFS : The Journalling Flash File System", *http://sources.redhat.com/jffs2/jffs2.pdf , 2001*

[14] Rosenblum M., Ousterhout J. K.: "The Design and Implementation of a Log-Structured File System", *ACM Transactions on Computer Systems, 10, 1, pp. 26-52, February 1992*

[15] ST Microelectronics, AN1822, Application note: "Wear Leveling in Single Level Cell NAND Flash-memories", *November 2004*

[16] Samsung, Application Note: "XSR1.5 Wear Leveling", *May 2007*

[17] SanDisk Corporation, White Paper: "SanDisk Flash-memory Cards Wear Leveling", *Doc. No. 80-36-00278, October 2003*

[18] Chang Li-Pin: "On Efficient Wear Leveling for Large-Scale Flash-MemoryStorage Systems", *Proceedings of the 22nd ACM Symposium on Applied Computing, 2007*

[19] M. L. Chiang, Paul C. H. Lee, R. C. Chang, "Using Data Clustering To Improve Cleaning Performance For Flash Memory", *Software - Practice and Experience, 1999*

[20] JFFS2, http://sourceware.org/jffs2/

[21] Aleph One Company, Cambridge, UK: "Yet Another Flash File System", *http://www.aleph1.co.uk/yaffs/index.html, 2002*

[22] Chang Y.-H., Hsieh J.-W., Kuo T.-W.: "Endurance Enhancement of Flash-Memory Storage, Systems: An Efficient Static Wear Leveling Design" *Proc. 44th ACM/IEEE Design Automation Conference (DAC) '07, 212-217, 2007*

[23] Samsung Electronics Co., Application Note: "NAND Flash ECC Algorithm (Error Checking & Correction)", *June 2004*

[24] Micron, Technical Note TN-29-08: "Hamming Codes for NAND Flash-memory Devices Overview", *May 2007*

[25] Chen S., Spansion, Application Note: "What Types of ECC Should Be Used on Flash-memory?", *November 2007*

[26] Mohammad, M. G., Saluja, K. K., Yap, A.: "Testing flash-memories" *Proc. 13th International Conference on VLSI Design, 2000, 406-411*

[27] Mohammad, M. G., Saluja, K. K.: "Flash-memory disturbances: modeling and test" *Proc. 19th IEEE on VLSI Test Symposium VTS 2001, 2001, 218-224*

[28] Chiu Sau-Kwo, Yeh Jen-Chief, Huang Chih-Tsun, Wu Cheng-Wen: "Diagonal Test and Diagnostic Schemes for Flash-memories" *Proc. International Test Conference, 2002, 37-46*

[29] Cheng, K.-L.; Yeh, J.-C.; Wang, C.-W.; Huang, C.-T. & Wu, C.-W.: "RAMSES-FT: a fault simulator for flash-memory testing and diagnostics" *Proc. 20th IEEE VLSI Test Symposium, 2002, 281-286*

[30] Kelly L. Hirsch, Data I/O Corporation, TechnicalGuide: "Programming NAND devices", *2003*

[31] Mohammad M. G., Saluja K.K., Yap A.: "Fault Models and Test Procedures for Flash-memory Disturbances", *Journal of Electronic Testing: Theory and Applications, Volume 17, Issue 6, December 2001 , pp 495 – 508, 2001, ISSN:0923-8174*

[32] Xin, Y., Ming, R. C., Xiong, H. B.: "A Flexible Garbage Collect Algorithm for Flash Storage Management", *Proc. Second International Conference on Future Generation Communication and Networking FGCN '08, 2008, 1, 354-357*

[33] T. Blackwell, J. Harris, and M. Seltzer, "Heuristic Cleaning Algorithms in Log-Structured File Systems", *Proceedings of the 1995 USENIX Technical Confer-ence, pp. 277-288, 1995*

[34] Sheng-Jie Syu, Jing Chen, "An Active Space Recycling Mechanism for Flash Storage Systems in Real-Time Application Environment", *Proc. of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'05), 2005*

[35] Samsung, Application Note: "XSR1.5 Bad Block Management", *2007*

[36] IEEE Standards Dept.: "IEEE 1005 Standard Definitions and Characterization of Floating Gate Semiconductor Arrays", *1999*

[37] Yeh, J. C.; Cheng, K.-L.; Chou, Y.-F. & Wu, C.-W.: "Flash Memory Testing and Built-In Self-Diagnosis With March-Like Test Algorithms", *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, 2007, 26, 1101-1113*

[38] Wang, W.-L., Song, Z.-W.: "An automatic design for flash memory testing", *Proc. IEEE International Workshop on Memory Technology, Design and Testing MTDT 2007, 2007, 46-49*

[39] Huang, C.-T., Yeh, J.-C., Shih, Y.-Y.: Huang, R.-F., Wu, C.-W: "On test and diagnostics of flash-memories", *Proc. 13th Asian Test Symposium, 2004, 260-265*

[40] Bernardi P., Rebaudengo M., Reorda M. S., Violante M.: "A P1500-compatible programmable BIST approach for the test of embedded flash-memories", *Proc. Design, Automation and Test in Europe Conference and Exhibition, 2003, 720-725*