Proc. of the Second Intl. Conf. on Advances in Applied Science and Environmental Engineering - ASEE 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-033-0 doi: 10.15224/ 978-1-63248-033-0-14

Performance and Solidity Schema for NAND Flash Memory-based Solid State Disk

AHMED I N ALSALIBI, PUTRA SUMARI

***Abstract* — significant attention has been paid to the Flash Memory based Solid State Drive (SSD) which made replacing the existing HDD, used as a storage unit across the world, very possible. Different from traditional disks, SSD uses semiconductor chips to store data. This structure enjoys very original technical characteristics including Low power consumption, shock resistance and high performance in random access. Those features can overcome the shortcomings of magnetic disks. Moreover, SSD is a complex storage system with its own features. Flash memory, the basic unit of SSD, has many distinctive characteristics that lead to various challenges. Flash memory doesn't support a write operation feature. A write operation can only be performed on an empty or erased unit which makes it more time-consuming. Besides, flash memory has a limited number of erase cycles. Moreover, each storage unit has limited number of erase cycles. In this paper, we propose an efficient schema called Performance and Solidity (PSS) to overcome the challenges, improve reliability and to enhance its performance by using wear leveling and organizing data based on access patterns. To prove the validity of PSS, both DiskSim and FlashSim simulators was used to evaluate the performance of the PSS and compare the results with other algorithms. Experimental results show that PSS performs better than the existing algorithms in terms of valid data distribution overhead and erase count distribution.**

Keywords* — *NAND Flash Memory, SSD, Wear Leveling

I. Introduction

Flash memory is a type of Read only memory which can be erased and reprogrammed electrically. It was created by professor, Fujio Masuoka, in the 1980s during his work in Toshiba Company. The memory being nonvolatile and its ability to keep data even when the power is off is what attracted Masuoka's attention the most. A typical flash memory structure consists of packages, with each one there is one or more dies (chips) sharing a single serial I/O bus and a number of common control signal.

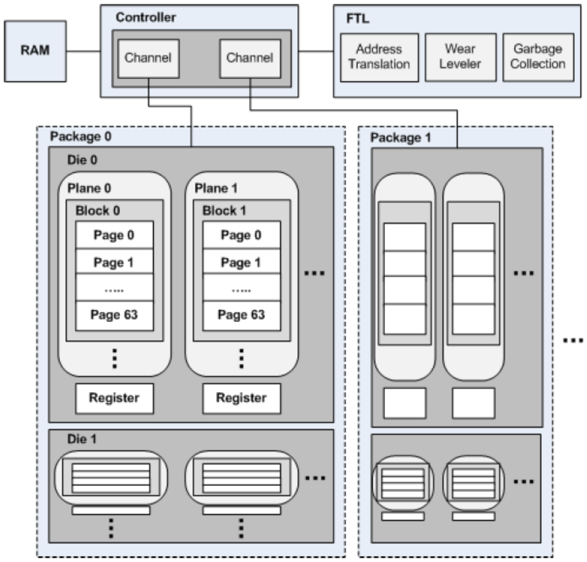


Figure 1. *typical Flash Memory Architecture*

Furthermore, each die consists of a number of planes, as shown in Fig. 1 plane also consists of many blocks which further consist of multiple pages with 2KB for each one. The size of the block may be 64 KB, 128 KB and 256 KB etc., based on manufacturer. Furthermore, each page contains two areas: Basic data area and a small spare Out-of-Band (OOB) which include multiple of information such as Error Correction Code (ECC), the Logical Page Number (LPN), erase counter and the page state, etc. [1].

Flash memory doesn't support the update feature; a write operation can only be executed after an erase operation which makes it more time-consuming. [2], [3]. One disadvantage of flash memory is that an erase operation. Unfortunately, erase operation should be done at block level instead of a page level, Block includes many pages. A write operation can be done only with an erased block. Therefore, to overwrite any page in a block, all the pages in the same block must be erased. Erase-before-write is the unique issue of the flash memory and is the main cause of the poor performance in flash memory [4], [5]. Each Flash memory has a limited number of erase operations before it becomes worn-out and invalid.

Before the erase operation, all the valid pages in the chosen block should be moved to other free blocks. When there are not enough free spaces, the flash memory will conduct a garbage collection operation. A block with invalid pages that doesn't need any complicated algorithms or

Ahmed I N Alsalibi
School of Computer Science, University Science Malaysia
Palestine

Putra Sumari
School of Computer Science, University Science Malaysia
Malaysia

procedures on order for it to be selected would be the best choice for an erase operation. Whereas blocks with a mixture of valid and invalid pages have to go through a selection process so they can be used efficiently. There is a need for an effective algorithm due to the limited number of erase operations in each block in order to ensure that blocks are not exhausted before being well-used.

II. Problem Statement

As a novel storage technology, the SSD provides not only promising features but also many critical challenges. On one hand, the physical semiconductor characteristic of SSDs brings high performance e.g. A low latency of 75 µs and high band width of 250MB/sec [6]. Additionally, SSDs are overall different from traditional HDD, which dominated the storage systems for decades. Efficient and effectively integrating SSDs, a novel storage technology with complicated and various features, into the existing storage structure and hierarchy require potential research efforts. In details, we need to address the following complicated and critical challenges:

- **Block Lifespan:** Each block in flash memory has lifespan (limited number of erase cycle) after this threshold, the block becomes unreliable, e.g. a multi-level cell (MLC) flash memory typically supports 10,000 erase cycles. If the specific block is erased and then reprogrammed every second, the block will skipped the 10,000 cycle limit in just three hours. To avoid this scenario, wear-leveling policy that wears out all blocks as evenly as possible is necessary [7].

- **Out-Place Updating:** Updating data in flash memory is implemented via the out-place policy rather the in-place policy, since in place policy requires a block to be erased first before the new updated data can be stored in the same location (pages). To avoid calling the erase operation every time data have been updated, the out-place-updating policy is implemented. New updated data is stored into a new location (pages) while the original copy is set as invalid (garbage).

- **Cleaning Process:** The garbage requires both reclaiming and then erasing. However, the cleaning operation is carried out by the erase operation and it is implemented at block level. The block may include valid data. And so, before starting the process, all valid data must be moved out into the available free space in other free blocks.

III. Related Work

In this section we discuss a series of research studies on the above-mentioned three critical problems. Many researchers tried to find out solutions regarding this issue. For instance, in the greedy (GR) method [8] the block with the highest number of invalid pages is most likely to be selected. This method is only applied in case of running out of free blocks. Blocks are kept in a queue based on the number of erase cycles in each block; the block with the lowest number will be at the front.

The block erasures cost for the steadly distributed data will be reduced because of this method, but when there is no harmony concerning average data access, then the performance will be poor while the operation cost will be on the rise. The Cost Benefit method is proposed [9]; to overcome the defects of GR method which does not take into consideration the number of erase cycle for blocks. The cost-benefit policy choses the most useful block according to the (1) for all active blocks during the block erasure operation. Variable (a) refers to the age of block since it was last updated and variable (u) refers to the block usage while age is the time since the latest update modulation. The cost for copying is represented by two terms, u in terms (2u) which represent read validation the block and u in terms (1-u) to write back them. Garbage collection procedure will select the block with the highest value. This method performance is better than GR method. But it does not take the wear level of blocks into consideration.

$$Costbenfit = \frac{a * (1-u)}{2u} \qquad (1)$$

Menon and Stockmeyerin [10], believe that it is very likely for the younger blocks to be rewritten or updated. Hence, older blocks will be selected for the erasure operation. Exclusively, the target will be those blocks that come to a certain age "age- threshold". By means of combining old blocks, the least used blocks will be given priority to be cleaned as they hold the largest amount of free space. The utilization of blocks is measured by the number of valid pages in the block. Blocks should wait until they reach an old-enough age in order to be selected by the garbage collection operation. The mechanism of this algorithm works in a way in which older blocks with fewer pages are in need of being updated extremely compared to younger blocks that have more pages. A calculation of an ideal age-threshold is made by tracing hot data frequency and the average of block utilization. The average of block utilization calculation is illustrated in (2).

Authorsin [11], introduced another solution to enhance the garbage collection performance by dividing block data into steady and non-steady. The main function of CAT policy is to minimize the erase operations and to evenly wear out the blocks in the flash memory. A fine-grained method is used by CAT policy to efficiently divide data into hot and cold in order to decrease over head of cleaning. Data blocks in the die can be categorized in three categories, depending on their stability: Read-only, cold data and hot data. Read only data refers to the data that has not been changed since it was created.

$$ABU = \frac{\text{TotalPages}}{\text{Number of Blocks}} * capacity \qquad (2)$$

The data that gets rarely updated is called the cold data, while the data that often gets updated is the hot data. All the valid data within a block are transferred to empty blocks in the flash memory after selecting a target block for cleaning. In order to choose the victim block, (3) is implemented by CAT policy. This equation includes the cleaning cost, and the aging time and number of erase cycles. (u) refers to the utilization level of valid data in a block whereas, (1-u) refers to cleaning cost of (u). The age refers to the period of time since the creation of the block. The blocks with the lowest value resulting from (3) are chosen as Target blocks.

$$\mathrm{CAT} = \frac{u}{1-u} * \frac{1}{age} * ct \qquad (3)$$

CATA algorithm introduced by [12], joins the age-sort method with the CAT policy. The blocks that will be selected for the erasure are those with the maximum value as calculated by the (4). Variable (u) refers to the block utilization; (a) refers to the age of block since its last update (e) refers to the number of erase cycle.

$$\mathrm{CATA} = \frac{u}{1-u} * \mathrm{a} * \frac{1}{e} \qquad (4)$$

IV. Performance and Solidity Schema

PSS gather blocks to two clusters: valid cluster (cluster 1) and invalid cluster (cluster 2). Valid cluster refers to the blocks, which have numbers of valid pages more than invalid pages. Invalid cluster refers to the blocks, which have numbers of invalid pages more than valid pages. Generally, invalidation will be faster for blocks which include hot data than the blocks that include cold data. Having hot pages with each other in the same blocks makes such blocks invalidated faster, increasing the possibility of these blocks to be chosen as target blocks in the future. However, if a hot block is chosen as the target, this means having most pages in invalid status. When the scarcity of free blocks inside SSD becomes more critical 15%, block selection method will be invoked to starting its function in a bid to produce more free blocks. Instead of randomly, this task will be done using organized procedure and based on the requirement of SSD in case more concern to performance the algorithm prefers the blocks that were stored in invalid cluster as a victim blocks using Victim Block Score (VBS) equation (6) which quote from Kim and lee equation [13]. Thus, it has few numbers of valid data to be moved to other free blocks before erase operation.

$$VBS = (1\text{-}\lambda) \left(\frac{\mathrm{valid}_x}{\mathrm{valid}_x + \mathrm{invalid}_x}\right) + \left(\frac{1}{\mathrm{age}}\right) \qquad (6)$$

Otherwise, in case the SSD is more concern to the wear leveling and the performance is not significant, the algorithm will choose the victim block from valid cluster using (7).

$$VBS = \lambda \left(\frac{\varepsilon_x}{\varepsilon_{\max}+1}\right) + \left(\frac{1}{age}\right) \qquad (7)$$

valid_x Refers to number of valid pages, $\mathrm{invalid}_x$ Refers to number of invalid pages and age refers to time since last update of the block. The measure of the wear is calculated based on the wearing of NAND flash memory inside SSD using the following equation;

$$\lambda = \begin{cases} \dfrac{2}{1+e^{\left(\frac{k\varepsilon}{\Delta\varepsilon}\right)}} & if\ \Delta\varepsilon \neq 0 \\ 0 & if\ \Delta\varepsilon = 0 \end{cases} \qquad (8)$$

$$\Delta\varepsilon = \varepsilon_{\max} - \varepsilon_{\min} \qquad (9)$$

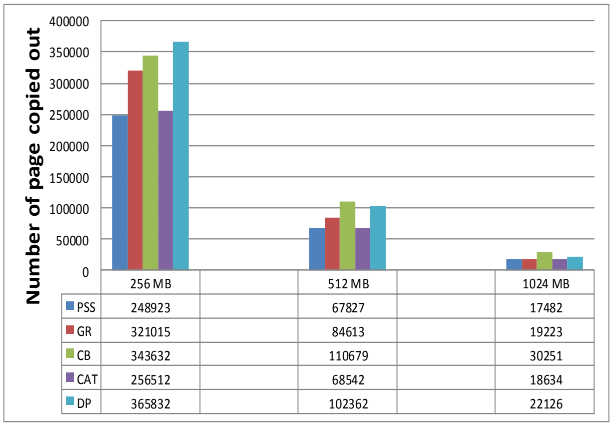
And is a measurement of the wear on SSD. Constant kε, is used to measure the responsiveness of SSD. Put differently, in case the wear begins rise, kε will calculate its speed of correction. From Trial and error, we have put kε=8 to provide a moderate response. (9) defines Delta Epsilon Δε, to be the variance between the block which has the highest erasure count ε_max, and block which has the least erasures count ε_min. Thus, when the wear of the flash is high, (7), which is more wear sensitive, will be determined as having more weight to be chosen to select a victim block than (6). However, if the wear λ is low, then the (6), which is more sensitive to performance will be weighted more heavily than (7). This permits the PSS to have a GC method that takes into consideration both performance and reliability based on the current condition of the flash memory. Another advantage of the Victim Block Score equation is the way that it intelligently regulates reliability and performance. the PSS garbage collection policy includes both policies in a weighted equation and is not in need of a predefined threshold to decide from which cluster the block should be selected. But rather, during each GC procedure, it intelligently adjusts the efficient balance of reliability and performance.

V. Results

In order to demonstrate the validation of PSS, we use both DiskSim [15] and FlashSim [16] simulators to evaluate the performance of the proposed schema. A threshold was predefined to invoke the garbage collection algorithm, since it is not efficient to have it run all the time. Accordingly, a counter is used to monitor the percentage of free blocks left in order to start the garbage collection algorithm. The garbage collection schema will be invoked when the amount of free blocks left in the flash memory is lower than 15% and stopped running when it reaches 20% of the overall memory. The trace files (workload) which used to evaluate the validity of the proposed schema are Financial1 and Financial2 which are I/O traces from OLTP applications [14] running at two large financial companies.

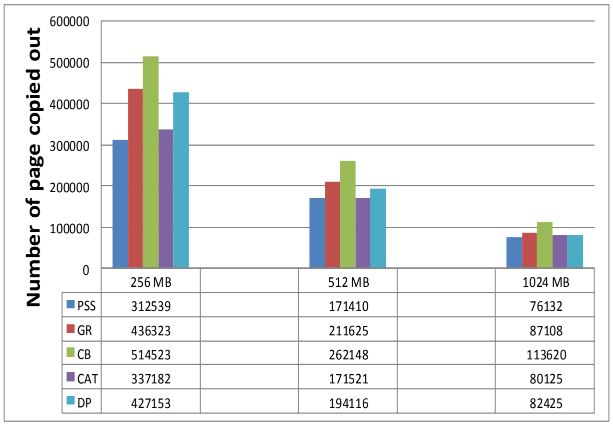
A. Valid Data Distribution Overhead

The performance of the garbage collection schema is determined by the number of the copied out pages as shown in Fig. 2 and 3. When the garbage collection is performed, the block selected to be erased, and the valid data (pages) in the block are then moved out to free pages in other blocks. Thus, if the victim block which is selected to be erased contains a huge number of valid data (Pages), the block recycling time also called overhead increases accordingly. As shown in both Fig. 2 and 3, PSS has a better performance than the GR algorithm in terms of valid data copied out. GR algorithm is expected to show better results in terms of valid data copied out. Because greedy algorithm selects the victim block which has the smallest number of valid pages. However, PSS perform better than GR in our experiments due PSS reorganized data to valid and invalid clusters, while GR does not take into consideration data classification. Although the major aim of data reorganization is enhancing the wear leveling, reducing the overhead of copy out is also considered. With PSS, gathering invalid data (blocks that have more invalid pages) together in the same cluster leads blocks to be more speedily invalidated faster, urging these blocks to have the greatest chance of being selected as victim block in the future. Selecting invalid block as a victim block, means having most pages in invalid status. As a result, a few valid pages will have to be copied out to other free pages.



	256 MB	512 MB	1024 MB
PSS	248923	67827	17482
GR	321015	84613	19223
CB	343632	110679	30251
CAT	256512	68542	18634
DP	365832	102362	22126

Figure 2. *Number of Pages Copied out for Different Algorithms (Financial 1)*

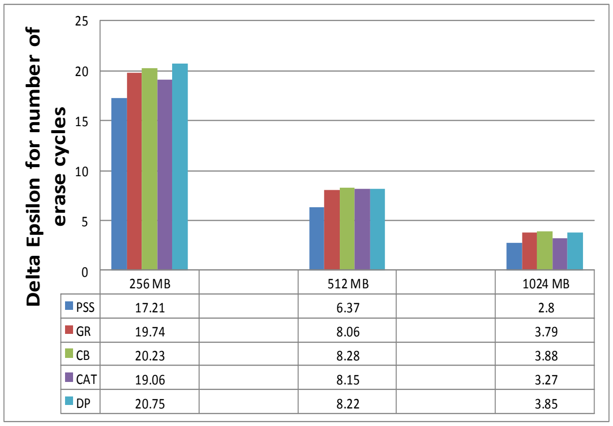


	256 MB	512 MB	1024 MB
PSS	312539	171410	76132
GR	436323	211625	87108
CB	514523	262148	113620
CAT	337182	171521	80125
DP	427153	194116	82425

Figure 3. *Number of Pages Copied out for Different Algorithms (Financial 2)*

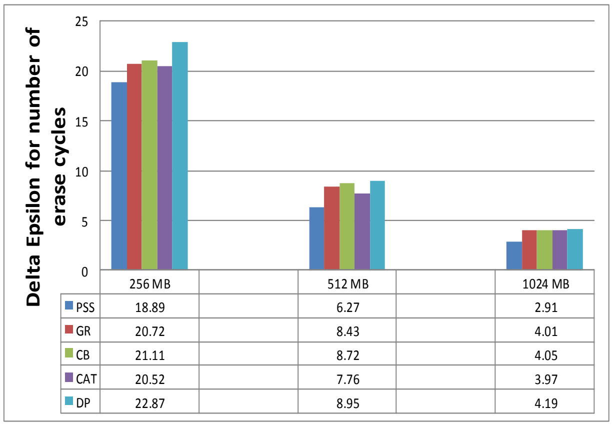
B. Erase Count Distribution

Fig. 4 and 5 shows the distribution of erase count for all the blocks in flash memory when different garbage collection schemas are used. Delta Epsilon is the variance number of erases cycle among the most worn block (the block which has the most number of erase cycle) and the



	256 MB	512 MB	1024 MB
PSS	17.21	6.37	2.8
GR	19.74	8.06	3.79
CB	20.23	8.28	3.88
CAT	19.06	8.15	3.27
DP	20.75	8.22	3.85

Figure 4. *Delta Epsilon for Number of Erase cycle for Different Algorithms (Financial 1)*



	256 MB	512 MB	1024 MB
PSS	18.89	6.27	2.91
GR	20.72	8.43	4.01
CB	21.11	8.72	4.05
CAT	20.52	7.76	3.97
DP	22.87	8.95	4.19

Figure 5. *Delta Epsilon for Number of Erase cycle for Different Algorithms (Financial2)*

least worn block (the block which has the least number of erase cycle). Thus, Delta Epsilon is used to measure wear leveling (reliability). Delta Epsilon presents how the garbage collection schemas erase each block evenly through the garbage collection procedure. The algorithm which has the smaller value of Delta Epsilon is a more reliable algorithm. As shown in Fig. 4, GR shows good performance when the Financial1 trace file is used. On the other hand, GR does not perform well when the Financial2 trace file is used as shown in Fig. 5. This is because a high number of write operations in the Financial2 and GR does not take into consideration the wear-leveling when choosing a victim block. Otherwise, as shown in Fig. 4 and Fig. 5, the Delta Epsilon for GR, DP and CB algorithm is high compared with CAT and PSS as those algorithms do not take into consideration the wear leveling of blocks. The CAT algorithm performs better as the data is redistributed in hot and cold blocks according to the modified frequency.

vi. Conclusion and Future Work

Flash memory has become popular in the trade market, but it has not yet gained credit to be the primary factor in enterprise environments because flash memory does not meet the requirements for both reliability and performance. Considerable research has been carried out in the fields of wear leveling and garbage collection. Many of existing algorithms focused mainly on performance with little consideration for reliability. However, in order for flash memory to have an applicable solution in an enterprise environment and data centers, it has to meet the requirements of performance and reliability. We proposed an efficient schema named PSS for flash memory to increase reliability, improving the wear-leveling, optimizing performance and enhance the efficiency of garbage collection. The PSS combines the victim block selection method with the efficient data reorganization method in order to reduce the cleaning cost. The simulation performance evaluation shows that the proposed schema PSS performs better than the existing algorithms in terms of the valid data reorganization overhead, erase count distribution. As a result, our objectives of this research have been achieved by improving the life span of flash memory in order for consumers to be able to enjoy the advantages of flash memory over HDDs. This is good news especially for enterprise environment and data centers which will greatly benefit from the extension of flash memory reliability and performance. For the future work we would like to take advantages of both SSD and HDD due the urgent need to appear efficient solution that has ability to combine performance and capacity needs within the budget limitation of the IT companies and organizations.

References

[1] E. Gal, S. Toledo, “Algorithms and data structures for flash memories,” ACM Computing Surveys (CSUR) 37 (2) (2005) 138–163.

[2] Y. Wang, D. Liu, M. Wang, Z. Qin, Z. Shao, Y. Guan, “Rnftl: a reuse-aware nand flash translation layer for flash memory,”ACMSigplan Notices 45 (4) (2010) 163–172.

[3] L. M. Grupp, J. D. Davis, S. Swanson, “The bleak future of nand flash memory,” in: Proceedings of the 10th USENIX conference on File and Storage Technologies, USENIX Association, 2012, pp. 2–2.

[4] Y. Yan, W. Chen, R. Fan, X. Guo, H. Guo, F. Zhang, L. Ding, K. Zhang, D. Lin, Y. Wang, “Analysis of the tid induced failure modes in nor and nand flash memories,”IEEE Transactions on Nuclear Science, 60, 224-229.

[5] Z. Sun, X. Chen, Y. Zhang, H. Li, Y. Chen,“Nonvolatile memories as the data storage system for implantable ecg recorder, ” ACM Journal on Emerging Technologies in Computing Systems (JETC) 8 (2) (2012) 13.

[6] Intel. Intel X25-E extreme SATA solid-state drive. http://www.intel.com/design/flash/nand/extreme, 2008.

[7] Amir Rizaan, Abdul Rahiman, and Sumari Putra,“Cleaning Process with Efficient Allocation Scheme Improves Flash Memory Performance,” Journal of Computers 7.3 (2012): 810-818.

[8] Wu, Michael, and Willy Zwaenepoel,“eNVy: a non-volatile, main memory storage system,”ACM SigPlan Notices. Vol. 29. No. 11. ACM, 1994.

[9] Kawaguchi, Atsuo, Shingo Nishioka, and Hiroshi Motoda,“A Flash-Memory Based File System,” In USENIX, pp. 155-164. 1995.

[10] J. Menon, L. Stockmeyer,“An age-threshold algorithm for garbage collection in log-structured arrays and file systems,” in: HighPerformance Computing Systems and Applications, Springer, 1998, pp. 119–132.

[11] Chiang, M-L., and R-C. Chang,“Cleaning policies in mobile computers using flash memory,” Journal of Systems and Software 48, no. 3 (1999): 213-231.

[12] Han, Longzhe, Yeonseung Ryu, and Keunsoo Yim,“CATA: a garbage collection scheme for flash memory file systems,” In Ubiquitous Intelligence and Computing, pp. 103-112. Springer Berlin Heidelberg, 2006.

[13] Kim, Han-Joon, and Sang-Goo Lee,“An effective flash memory manager for reliable flash memory space management,” IEICE Transactions on Information and Systems 85, no. 6 (2002): 950-964.

[14] http://traces.cs.umass.edu/index.php/Storage/Storage, accessed June. 2010.

[15] Buch, John S., et al,“The DiskSim simulation environment version 4.0 reference manual,” (2008).

[16] Kim, Youngjae, et al, “Flashsim: A simulator for nand flash-based solid-state drives,” Advances in System Simulation, 2009. SIMUL'09. First International Conference on. IEEE, 2009.

Ahmed Alsalibi received the B.S. degree in Computer Systems Engineering from Palestine Technical College, Gaza Strip, Palestine, in 2009, and the M.S. degree in Computer Science from University Science Malaysia. He is currently working toward the Ph.D. degree in Computer Science at University Science Malaysia. His research interests include Hybrid Storage Systems Design and Operating Systems.

Putra Sumari received the B.S. degree in Computer Science from University Science Malaysia, Penang, Malaysia, the M.S. degree, and his Ph.D. degree, both in Computer Science from University of Liverpool, U.K. Now he is a Associate Professor in the School of Computer Science (USM). His research interests include Video on Demand Application and Storage Systems.