

## **Implementing of Global Adaptive Algorithm in Read Write Access of Flash Storage Systems by managing Spatial and Temporal Localities**

**S.M Shamsheer Daula**

*Research Scholar [PP.ECE.18], Rayalaseema University, A.P-, India.  
Assistant Professor, Dept of ECE G Pulla Reddy Engineering College (Autonomous)*

**Dr. K E Sreenivasa Murthy**

*Professor & Head, G Pullaiah College of Engineering, Kurnool,  
Andhra Pradesh, India.*

### **Abstract**

The performance of embedded systems are observed by Design and efficiency of a system always maintain a trade-off relation. The data cache level have come with an introduction of reducing the addressing time to locate the effective storage, which directly shows a proportionality in time and energy access. However, at stages of mapping the level of expected speed reduces due to overwhelmed page clusters or repeated called and pending access layers. Many replacement approaches have shown their essence in improvising the mapping standards. This work has three level of presentations, the first and second choose a choice of prominent algorithm in mapping speed with symmetric reading and writing speeds in both spatial & temporal localities and other to show energy reduction by showing speed enhancement. The techniques are near to precise in lowering and minimizing the access times of addressing during the cache handling. This allows suitable energy reduction with very least performance challenges. The results are given with comparisons of various algorithms and energy utilization.

**Keywords:** APCLRU, CCFLRU, GASST Clock selection, FTL, TLB, NAND Flash memory, Control Block, Advanced Processors.

## I. INTRODUCTION

The performance of embedded systems are observed by many aspects. One of the most important method is memory management. Various framing methodologies have been proposed to explore the comparisons among design and performance evaluation of effect of memory in increasing the efficiency of the advanced embedded processors.[8][9] Some variations are been observed in current processors due to their deterministic nature. The methods studied in the proposed representation have advantages which are suitable for different application fields. In comparative cases, their usage challenges the performance of the WCET worst-case execution time estimates, which is mandatory in real-time systems to prove that the timing constraints of the system are met. For some applications with high reliability requirements, e.g., in Telecommunication switching, it is not suitable to identify erroneous data only at the time, when the data or information are explicitly needed. In contrast, degrading factors for the read and write should be detected as early as possible to allow for recovery before the data are requested by the system. Flash memory is equipped with a write buffer to improve its write performance. [12]. A Flash Translation Layer of cache is integrated into the Flash based memory controller [1] [4] [7]. However, when it comes to the concept in the flash translation layer in the given representation can be any kind of block-mapping- or hybrid-mapping- based FTLs, such as BIST-aided scan test and FAST.

## II. RELATED WORK:

The figure 1 represents the cached and non-cached arrangements of the accessible memory section. It is given with an intention by the authors that memory paging and access mechanisms stand as the basic knowledge for the untold explanations.

### 1. Initial configuration

Block 1000 (data)		Block 2000 (free)	
PPN	data	PPN	data
0	x	0	
1	y	1	
2	z	2	
3		3	

### 2. Writing a page

Block 1000 (data)		Block 2000 (free)	
PPN	data	PPN	data
0	x	0	
1	y	1	
2	z	2	
3	x'	3	

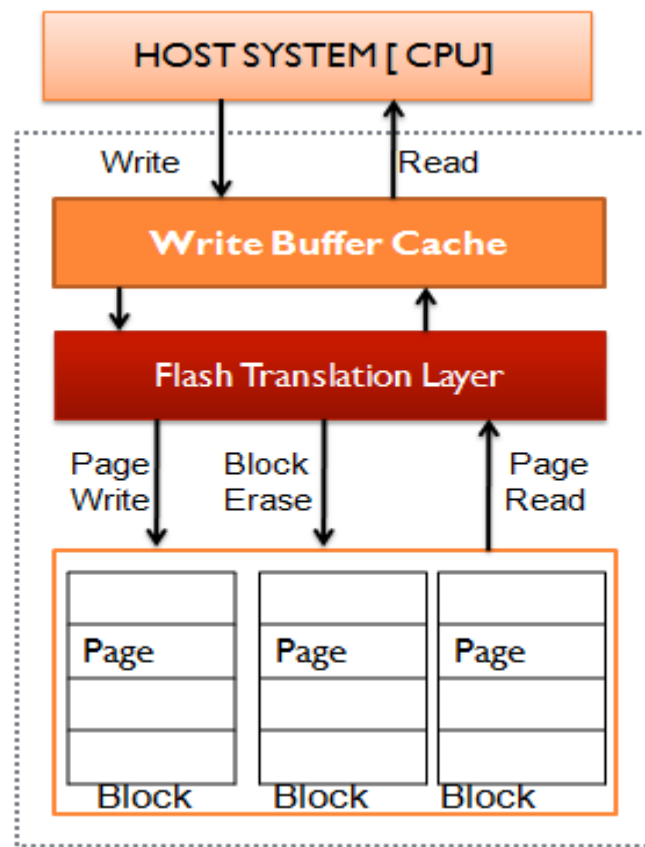
### 3. Erasing a block (garbage collection)

Block 1000 (free)		Block 2000 (data)	
PPN	data	PPN	data
0			
1		1	y
2		2	z
3		3	x'

Figure 1 FTL work on GC and Write.

Plenty of management methodologies have been proposed [11], [19] on virtual memory regarding for Flash-based memories in embedded systems. On the basis of providing secondary memories with respect to devices like hard disks, many approaches including optimal method, stack method and least and most significant recently used page method, show their effect [2],[5][10]. The least recently used (LRU) algorithm when compared is mostly used as the replacement algorithm. However, when Flash memory is applied as the secondary storage, an asymmetry arises in read and write operations. The time taken in representing a read operation is far less when stated in a row of comparison with write operation. When a write operation is initiated the availability of empty spaces is more required, so basically a search for empty block or locality is done and if it suffers through any unavailability then clearing of any level of pages or cluster or block is done. The approaches like buffer aware algorithm, clean first less used algorithm have their own benefits and flaws but they improve the system performance by reducing the number of write activities on the Flash memory. The write buffer to make it not only improve the performance but also work seamlessly with the management approach of virtual memory. Finally, the performance of these two designed approaches is evaluated.

**III. PROPOSED WORK:**



**Figure 2: Flash Storage Block Logic.**

Temporal locality tells that a called page may be reused once again and the pages related to the called page could be mapped in the near future, their reference is held in spatial locality. A figure 2 above suggest the alternate block mode [ABM] to manage the spatial and temporal access localities.[3],[17],[12],[14]. A global adaptive symmetry of spatial and temporal management [GASSST] methodology, is proposed to manage the ABMs. The logical arrangement in shown made possible by introducing the suitable multiplexers at required levels.[21],[22],[9] Same as CFLRU, this approach sets the window size statically. It is shown bypre-executingall the applications while setting static or dynamic window size from 10% to 100% of virtual memory and identify the best results for all the target applications.

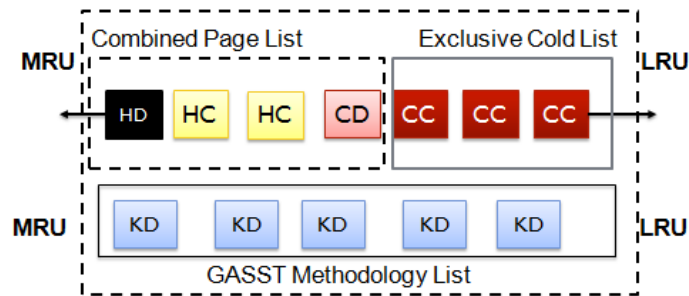
#### Algorithm

- 1: **APCLR** $U(P_i, data)$
- 2: **if** buffer cache has free area **then**
- 3: *Add\_To\_Write\_Buffer* $(P_i, data)$ ;
- 4: **else**
- 5: *Victim\_Cluster\_Select* $()$ ;
- 6: *Add\_To\_Write\_Buffer* $(P_i, data)$ ;
- 7: **end if**

The result is selected with the minimal additional overheads on Flash memory as the best result. The additional overheads include erase operations, read operations, and copy operations during garbage collection the Flash memory. With this approach, a suitable window size for the target application can be identified.

#### IV. METHODOLOGY

For the description of the proposed scheme, it is denoted with  $n$  the number of stages of the ABM that can perform ones complement addition and with  $k$  the number of bits of the RAM word (hence  $k < n$ ). The purpose of the proposed scheme is to assure consecutive block elements, are complementary [13]. Improve the performance of Flash-memory-based systems. Unlike previous work, where virtual memory and write buffer managements are designed separately, this paper proposed a new replacement algorithm for virtual memory, which cooperates with the write buffer and reorders the write sequences sent to the write buffer.



**Figure 3: GASST Methodology**

A replacement algorithm on the buffer management of write operations was also proposed to work effectively with the proposed approach in virtual memory.

This rises an issue of more load on performance treated as penalty to the performance [15][16]. When it makes a greater difficulty in finding an empty space but the padding methodology shows some improvising effect. When page in the ABM is available to be processed while the needed location is away, a side route method helps the information buffer and send the pages to the write buffer, APCLRU works seamlessly with WBS since WBS may prematurely evict dirty pages from the virtual memory to the write buffer.[9],[12] GASST has better I/O performance (measured in milliseconds) than BPLRU and CLC in most cases. WBLRU shows significant improvement over LRU, CFLRU, and CFDC. Some other approaches, such as CCFLRU [22] and LRU-WSR [19], are orthogonal to the proposed approach.

**Proposed Algorithm:**

**FUNCTION 1: ADD\_TO\_Control\_BUFFER**

```
( $P_i$ , data)
if  $C_i \subseteq C S$  then
    Write  $P_i$  to the cluster;
```

Move this cluster to MRU control position;

**end if**

$chance\_tag \leftarrow$  FALSE;

**FUNCTION 2: Control\_CLUSTER\_SELECT()**

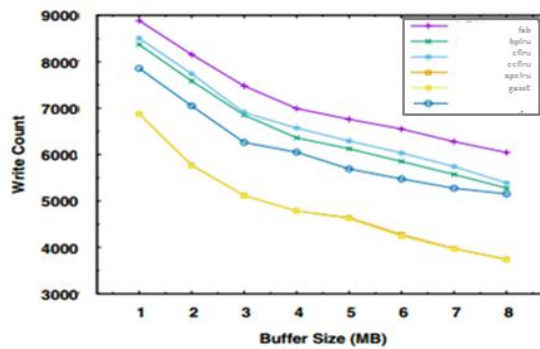
```
1:  $C_i \leftarrow$  the APCLRU cluster in the write buffer; 2: while  $C_i$  do
3: if  $C_i$  is large cluster  $\_chance\_tag$  is set then 4: Flush  $C_i$  to Flash memory;
5:  $C S \leftarrow C S - C_i$ ; return  $C_i$ ; 6: else
7:  $chance\_tag \leftarrow$  TRUE;
```

8:  $C_i \leftarrow$  the next APCLRU cluster; 10: **end if**  
 11: **end while**

The figure 3 shown above the flow of the references made under each of the control block. The portion of the SPM that remains unallocated after the pinned code section has been loaded constitutes the execution buffer. The execution buffer resembles a software cache for paged code.[21][22].

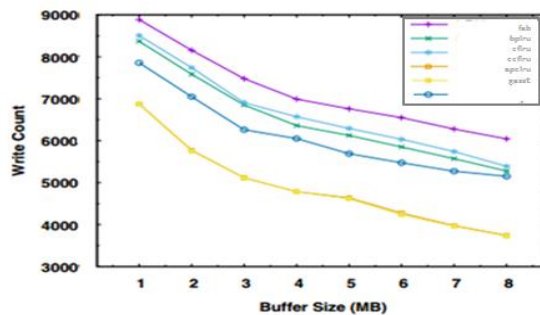
**V. WORK ANALYSIS**

At selected sizes of memories the given results out show the Comparisons and staged results for the algorithms at size of 16 MB data cache [23].It is assumed to be a hybrid mapping approach as selected in figure 3, as it has better tradeoffs between the mapping table size and Flash into consideration the correlation between write patterns and merge operations in hybrid FTL,The minimal allocation unit is thus one page, however, paged functions may consist of multiple pages. It loads the requested function from external memory into the execution buffer.



**Figure 4:** Comparisons at various hit ratios

The entry in the page codetable is changed to the function residing in the execution buffer by creating a new control block the goal can be enhanced algorithm proposed is to organize the write buffer in a way that produces suitable write patterns for FTL to increaseThe minimal allocation unit is thus one page, however, paged functions may consist of multiple pages.



**Figure 5:** Write Count Reduction Analysis

The minimal allocation unit is thus one page, however, paged functions may consist of multiple pages. It loads the requested function from external memory into the execution buffer in no of erase counts prominently.

**Table I:** Performance appraisal of hit ratios

Algorithm --- Buffer Size	2 MB	4 MB	6 MB	8 MB
FAB	14	18	20	24
BPLRU	16	18	22	24
CFLRU	29	39	45	52
CCFLRU	29	40	47	53
APCLRU	29	40	47	53
GASST	31	45	53	66

The tabulated results are obtained by the calculation of some fixed blocks and variable cluster sizes. The results propose an approximate elevation in the performance of the new methodology, with increase in efficiency with increase in the size of memory. Table II attributes are performed with experimental arrangement by loading blocks with address and data values and their linked localities are deviated from block to block and in other case the cluster link is enhanced putting emphasize on how could be access speed in the linked clusters to justify the spatial locality access.

**VI. CONCLUSION**

With an overall idea of improving the memory management in flash based embedded systems, this paper focusses on an approach to enhance the access time by reducing the miss ratio and increasing the hit ratio with more and fast clearances in the cluster and block regions. Besides justifying the earlier methods and embarking the improvements over them is emphasized. The complete write buffer management suits the new approach in minimizing the delay time spent only to find a free write space. The FTL logic is seamlessly addressed through control block management. The theme point is sorted out experimentally as to maintain symmetry in the read and write speeds the flash storage systems.

## REFERENCES

- [1] T. Parker, J. Zang, Kang, J.-S. Kim, and J. Lee, "WBLRU: A replacement algorithm for flash memory," in *Proc. Int. Conf. Compil., Arch. Synth. Embed. Syst.*, 2008, pp. 234–241.
- [2] L. Shi, J. Li, C. J. Xue, C. Dang, and X. Mhou, "CFLRU: A unified write cache management for flash memory," in *Proc. ACM Int. Conf. Embed. Softw.*, 2011, pp. 339–348.
- [3] J. Mik, J. M. Kate, S. Hon, S. L. Min, and Y. Cho, "A space-efficient flash translation layer for compact flash systems," *ACM Trans. ComputeElect.*, vol. 48no. 2, pp. 366–375, May 2010.
- [4] S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S. Park, Song, "A log buffer-based flash translation layer using fully-associative sector translation," *IEEE Trans. Embedded. Syst.*, vol. 6, no. 3, pp. 1–12, 2007.
- [5] S. Kang, S. Park, H. Jung, H. Shim, and J. Cha, "Performance trade-offs in using NVRAM write buffer for flash memory-based storage devices," *IEEE Trans. Comput.*, vol. 58, no. 6, pp. 744–758, Jun. 2009.
- [6] 7. J. Seol, H. Shim, J. Kim, and S. Maeng, "A buffer replacement algorithm exploiting multi-chip parallelism in solid state disks," in *Proc. Int. Conf. Compil., Arch., Synth. Embed. Syst.*, 2009, pp. 137–146.
- [7] X.-Y. Hu, E. Eleftheriou, R. Haas, I. Iliadis, and R. Pletka, "Write amplification analysis in flash-based solid state drives," in *Proc. SYSTOR: Israeli Experim. Syst. Conf.*, 2009, pp. 1–9.
- [8] Shamsheer Daula, Dr K E Sreenivasa Murthy "Assymmetric Read/ Write memory management" International Conference, IEEE Section, A.P, 2014.
- [9] O. Neil, J. Swift, G. Lenz "Flash based storage management with cache handling" in *IEEE Trans. Comput.*, vol. 8, no. 46, pp. 74–78, Feb. 2009.
- [10] 7. J. Seol, H. Shim, J. Kim, and S. Maeng, "A buffer replacement algorithm exploiting multi-chip parallelism in solid state disks," in *Proc. Int. Conf. Compil., Arch., Synth. Embed. Syst.*, 2009, pp. 137–146.
- [11] S.-Y. Park, D. Jung, J.-U. Kang, J.-S. Kim, and J. , in *Proc. Int. Conf. Compil., Arch. Iee*, "CFLRU: A replacement algorithm for flash memory," in *Proc. Int. Conf. Compil., Arch. Synth. Embed. Syst.*, 2006,
- [12] L. Shi, J. Li, C. J. Xue, C. Yang, and X. Zhou, "EXLRU: A unified write buffer cache management for fmemory," in *Proc. ACM Int. Conf. Embed. Softw.*, 2011 pp. 339–348.
- [13] J. Kim, J. M. Kim, S. Noh, S. L. Min, and Y. Cho, "A space-efficient flash translation layer for compact flash systems," *IEEE Trans. Consumer Electron.*, vol. 48, no. 2, pp. 366–375, May 2002.
- [14] S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S. Park, and H.-J. Song, "A log buffer-based flash translation layer using fully-associative sector translation," *ACM Trans. Embed. Comput. Syst.*, vol. 6, no. 3, pp. 1–18, 2007.
- [15] S. Kang, S. Park, H. Jung, H. Shim, and J. Cha, "Performance trade-offs in using NVRAM write buffer for flash memory-based storage devices," *IEEE Trans. Comput.*, vol. 58, no. 6,



- [16] D. Seo and D. Shin, "Recently-evicted-first buffer replacement policy for flash storage devices," *IEEE Trans.Consumer Electron.*, vol. 54, no. 3, pp. 1228–1235, Aug.2008.
- [17] J. Seol, H. Shim, J. Kim, and S. Maeng, "A buffer replacement algorithm exploiting multi-chip parallelism in solid state disks," in *Proc. Int. Conf. Compil., Arch., Synth.Embed. Syst.*, 2009, pp. 137–146.
- [18] X.-Y. Hu, E. Eleftheriou, R. Haas, I. Iliadis, and R. Pletka, "Write amplification analysis in flash-based solid state drives," in *Proc. SYSTOR: Israeli Experim. Syst. Conf.*, 2009,
- [19] H. Jung, H. Shim, S. Park, S. Kang, and J. Cha, "LRU-WSR: Integration of LRU and writes sequence reordering for flash memory," *IEEE Trans. Consumer Electron.*, vol. 54, no. 3,
- [20] Y. Ou, T. Härder, and P. Jin, "CFDC: A flash-aware replacement policy for database buffer management," in *Proc. 5th Int. Workshop Data Manage. New Hardw.*, 2009,
- [21] L. Shi, C. J. Xue, and X. Zhou, "Cooperating write buffer cache and virtual memory management for flash memory based systems," in *Proc. 17th IEEE Real-Time Embed.Technol. Appl. Symp.*, Apr. 2011, pp. 147–156.
- [22] Y. Ou, T. Härder, and P. Jin, "CFDC: A flash-aware replacement policy for database buffer management," in *Proc.5th Int. Workshop Data Manage. New Hardw.*, 2009, pp. 15–20.
- [23] L. Shi, C. J. Xue, and X. Zhou, "Cooperating write buffer cache and virtual memory management for flash memory based systems," in *Proc. 17th IEEE Real-Time Embed.Technol. Appl. Symp.*, Apr. 2011, pp. 147–156.

