ISSN 2348-375X

Unique Journal of Engineering and Advanced Sciences

Available online: www.ujconline.net

Research Article

# OPTIMAL PAGE ALLOCATION OF HYBRID MAIN MEMORY FOR TASK ALLOCATION ON NON VOLATILE MEMORY

Saravanapriya SK[1*], Arulanantham D[2]

[1]PG Student, Department of Electronics and communication Engineering, Nandha engineering college, Erode-638052, Tamilnadu, India.
[2]Assistant Professor, Department of Electronics and communication Engineering, Nandha engineering college, Erode-638052, Tamilnadu, India.

Received: 29-12-2013; Revised: 27-01-2014; Accepted: 26-02-2014

*Corresponding Author: **Saravanapriya S.K**
PG Student, Department of Electronics and communication Engineering, Nandha engineering college, Erode-638052, Tamilnadu, India. Email:s.chaaru@yahoo.in

## ABSTRACT

Dynamic random access memory (DRAM) has been located in the main memory of computer architecture for the last several decades. Here new memory technologies such as Phase Change RAM (PRAM), Ferroelectric RAM (FRAM), and Magnetic RAM (MRAM) have been suggested; these can provide more memory capacity and less energy consumption than DRAM. In the proposed work, efficient task selection in DRAM and PRAM by means of Optimal Page Allocation in Main Memory architecture is considered.

**Keywords:** Dynamic random access memory (DRAM), Phase Change RAM (PRAM), Page allocation.

## INTRODUCTION

Main memory has become a significant energy consumer, contributing to as much as 30-40% of total consumption on modern server systems. In this paper, we propose a new approach for tackling the high levels of energy dissipation in main memory while minimizing their impact on performance. We introduce a new heterogeneous organization for main memory that is composed of DRAM and PRAM memories[1-6]. Dynamic random access memory (DRAM) has large portion of consumed energy. About 30% of energy is consumed by DRAM main memory. The reason is that DRAM always spends the energy to keep stored data even when it does not work because of its volatile characteristics. In recent years, Phase Change RAM (PRAM) technology has been researched to replace DRAM.PRAM is one of the promising candidates for future main memory because it is a byte-addressable and non-volatile memory. The properties of PRAM that we leverage are its lower read access and standby power compared to DRAM while having a comparable throughput. However, the primary challenges in using PRAM include its lower write and the higher power cost of write accesses compared to DRAM. These properties motivate the use of heterogeneous memory architecture consisting of both DRAM and PRAM, which we refer to as PDRAM, enabling exploitation of positive aspects of the respective memories[7-10].

## PROBLEM OVERVIEW

### 1) Background

PRAM differs from flash in several important characteristics, both in terms of scaling (i.e., higher densities and potentially lower costs/bit) and endurance[11-13]. There are a variety of estimated endurances for PRAM, possibly because there is some uncertainty as to which chalcogenic material yields the best tradeoffs[14], and also because the endurance for a large array is expected to be lower than for a single bit[14]. It seems reasonable to assume an endurance of 10^8, which is 2 or 3 orders of magnitude better than flash, but still finite. PRAM also eliminates the erase phase associated with flash, greatly simplifying the overall system design. It appears that PRAM writes cannot be faster than about 100ns[14], which implies that it might still be necessary to buffer writes in DRAM and reads to happen at DRAM-like speeds is expected, however. Thus, PRAM has larger peripherals and consumes more write energy than DRAM.

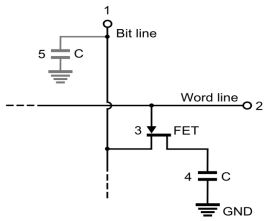



**Figure 1: DRAM cell**

DRAM uses memory cells consisting of one capacitor and one transistor to store each bit as shown in figure 7.3. This is the cheapest and highest in density, so it is used for the main memory in computers. However the electric charge that stores the data in the memory cells slowly leaks off, so the memory cells must be periodically refreshed (rewritten), requiring additional circuitry.FPM DRAM (Fast page mode DRAM) is an older type of asynchronous DRAM that improved on previous types by allowing repeated accesses to a single "page" of memory to occur at a faster rate. For several decades, general-purpose CPUs have used DRAM for main memory. DRAM has many good features, and has benefited from Moore's Law, but DRAM is not perfect: it is relatively expensive in power and cost, as a fraction of an entire computer, and it is hard to put enough of it near a CPU. These problems are especially pressing in "scale-out" server farms, where we want both increased server density and reduced heat density.On the other hand, flash memory can be denser, cheaper, and more power-efficient than DRAM, but it has problems with access timing, access unit sizes, and endurance. Because of these problems, and because it is reasonably non-volatile, flash is typically used in the storage hierarchy, rather than as main memory.

**2) Related work**

When using PRAM as main memory, recent work[8-10] introduces several efficient methods to reduce the write activity, for example, removing redundant write, row shifting and segment swapping or re-mapping. All these techniques are carried out in hardware circuit or memory controller level. In order to make the hybrid main memory possible[3], and propose hybrid hardware/software solutions. Dhiman *et al.* proposes a hybrid PRAM and DRAM main memory[3]. They design a memory controller at page level granularity (with OS management) with a page manager (using frequency to avoid hot spots), and also evaluate the performance overhead and energy saving of this architecture. Mogul *et al.* provide OS support for a hybrid NOR flash and PC-RAM main memory[4]. Anew architecture with a hierarchical and hybrid main memory for many-core system is proposed in [4]. Efficient management techniques are designed to achieve high performance and low energy consumption through migration of pages between DRAM and PRAM. In [6], runtime-adaptive time out control and DRAM bypassing methods are proposed to avoid DRAM refreshes to save power. Based on the application-specific properties of embedded systems[16], analyzes the application and reduces the number of writes by data migration and re-computation for non-volatile memories.

**OUR APPROACH**

In this paper, we re-consider the allocation of tasks in them. DRAM memory, putting variables in PRAM can save power, but may lead to many writes to the PRAM which will shorten the memory's lifetime and degrade the performance. Therefore, trade-offs should be investigated when considering different objectives jointly, such as power, performance and endurance. The focus of this paper is to take advantages of PRAM while trying to minimize its negative impacts. For different hybrid architectures, we propose different and novel algorithms to partition variables in different memory banks to reduce power consumption and the number of writes on PRAM.

Task allocation is done by means of Optimal Page Allocation on Main Memory (OPAMM). An evaluation framework termed OPAMM, which calculates optimal performance of the hybrid memory environment. As the existing systems gives controlled output for all unallocated pages in RAM memory and consumes energy even for the unused pages, a page allocation scheme is proposed as it would provide energy only to the needed pages and the pages to be used.

**CHALLENGING KEY ISSUE**

While allocating pages in the hybrid memory system, there are two methods which are static allocation and dynamic allocation. The difference between two methods is the acceptance of page migration between DRAM and PRAM. While finding optimal value of system's performance, consideration of page migration is treated as a challenging problem since the general memory allocation problem is NP-complete and static allocation is also complex enough. Because of the above mentioned reasons, previous researches focused on the static allocation to get the optimal values.

To find the exact optimal value of hybrid memory, consideration of the dynamic case is needed and it is very challenging issue as we mentioned before. To solve this problem, we design a method which calculates the optimal value of dynamic allocation case.It uses the concept of paging where, paging use fixed size pages as base unit, usually 4k large. It provides one virtual address space in the logical domain, which is mapped to physical memory through one mapping entry for each page in the virtual address space[15].

.

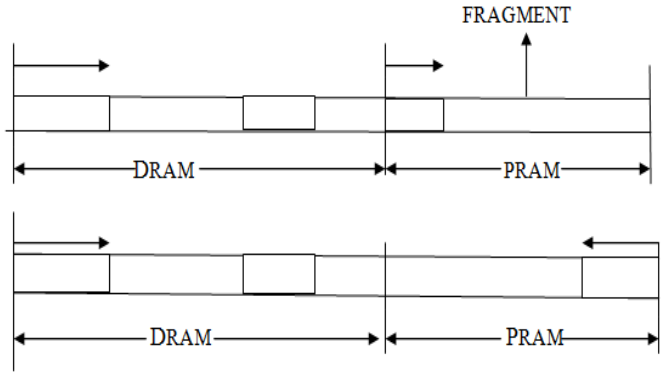



**Figure 2: Block diagram of task allocation**

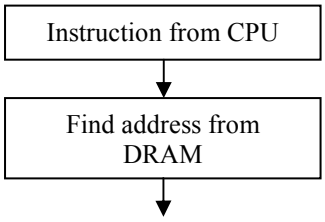
Thus page allocation is considered to be the main concept which uses the existing algorithms (1) for the allocation. The algorithms considers the size, arrival time of the task and the number of writes needed and hence it helps us achieving minimum PRAM writes, minimum PRAM size and minimum energy consumption. The objective is to exploit the advantages of DRAM and PRAM while hiding their disadvantages. However, such objectives are conflicting with the task allocation problem. For instance, minimizing energy consumption prefers to place all P-tasks into PRAM, the side effect of which is increasing the writes on PRAM. Therefore, careful balancing is crucial[16].

There are four constraints in total: 1) a threshold of energy consumption ($E_t$); 2) a threshold of PRAM size ($P_t$); 3) a threshold of DRAM size ($D_t$); and 4) a threshold of the number of writes on PRAM ($N_t$). By fixing any three of the constraints, we can optimize the fourth one. Thus, naturally four problems exist. Since it is similar to minimizing the PRAM size or the DRAM size, we only select to minimize the PRAM size.It presents the methodology to minimize energy consumption offline.First, all P-tasks and D-tasks are assigned in PRAM and DRAM, respectively. Thus, the energy consumption *E* will be the lower bound of this problem, while the used PRAM size P, used DRAM size D, and the number of writes on PRAM N might surpass the thresholds $N_t$, $P_t$, and $D_t$, respectively.

The heuristic MinN with the objective to minimize number of writes on PRAM offline, the principle of which is as follows. First, all tasks are assigned in DRAM. Thus, $D > D_t$ and $E > E_t$ might hold. Then, tasks need to be iteratively migrated from DRAM to PRAM until neither of these inequalities holds. At each iteration step, if R1 > R2, the task τi with the largest ($s_i/N_{wi}$ ) will be migrated from DRAM to PRAM. Else, (R1 ≤ R2), P-task with the largest ($|\_E_i|/N_{wi}$) will be selected for migration from DRAM to PRAM. When the iteration terminates, we check if $P \leq P_t$ holds. If yes, N will be the desired solution. For Case 3, the heuristic MinP with the objective to minimize the PRAM size offline is similar to MinN. Thus, MinP is omitted here.

Algorithm MinE (offline): Minimizing Energy Consumption
Require: Task set T = {$\tau_1$, $\tau_2$ . . .}, Nt, Pt, Dt.
Ensure: Energy consumption (E) of the hybrid memory.
1: Place all P-tasks in PRAM and all D-tasks in DRAM, Compute P, D, N;
2: R1 ← (N/Nt), R2 ← (P/$P_t$), R3 ← (D/Dt);
3: while R1 > 1 or R2 > 1 or R3 > 1 do
4: if R1 ≥ R2 and R1 ≥ R3 then
5: Migrate P-task $\tau_i$ with the smallest ($|\Delta E_i|/Nw_i$) from PRAM to DRAM;
6: end if
7: if R2 > R1 and R2 ≥ R3 then
8: Migrate P-task $\tau_i$ with the smallest ($|\Delta E_i|/s_i$) from PRAM to DRAM;
9: end if
10: if R3 > R1 and R3 > R2 then
11: Migrate D-task $\tau_i$ with the smallest ($|\Delta E_i|/s_i$) from DRAM to PRAM;
12: end if
13: Recompute R1, R2, R3;
14: end while
15: Compute E; 16: Return E;

The algorithm has been implemented in the page allocation scheme as shown in Figure 3.



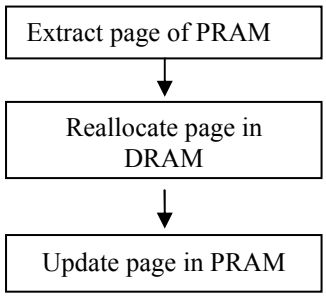




**Fig.3.Page allocation scheme**

1) Command from CPU: Since there are caches and write buffers between the CPU and memory, which will throttle the rate of writes. However, the analysis will help to estimate PRAM reliability under extreme cases. From this RAM receives the command from CPU.
2) Find Addresses in DRAM: Address information from memory is obtained. This has the page information of DRAM memory.
3) Extract page of PRAM: Page information in PRAM memory is acquired. Thus the pages of PRAM are combined with the DRAM memory.
4) Reallocate page in DRAM: With the page information of PRAM memory of DRAM is reallocated.
5) Update page in PRAM: Finally the page allocation is updated in PRAM memory. This gives hybrid combination of DRAM and PRAM. This was done by match finding in DRAM and PRAM page allocation.

## RESULTS AND DISCUSSION

The method is tested with simulation results using XILINX. **Xilinx Platform Studio (XPS) -** provides an integrated environment for creating software and hardware specification flows for embedded processor systems based on MicroBlaze and PowerPC processors. It also provides an editor and a project management interface to create and edit source code.

In this section, we present the parameter sensitivity study of the proposed heuristics. We conduct experiments on the task set. For each heuristic, there are three inputs. By fixing two inputs and changing the third one, we can derive different outputs. No solutions can be derived from MinE, MinE and MinE. The on-chip power consumption can be drawn as shown in figure 4 and 5. In each group of experiments, our method proceeds by fixing two parameters and changing the remaining one. $D_t$ has the largest impact on P.

**Table 1: Specifications of Target System**

| Parameter | DRAM | PRAM |
|---|---|---|
| Row read power | 210mW | 78mW |
| Row write power | 195mW | 773mW |
| Actual power | 75mW | 25mW |
| Standby power | 90mW | 45mW |
| Refresh power | 4mW | 0mW |

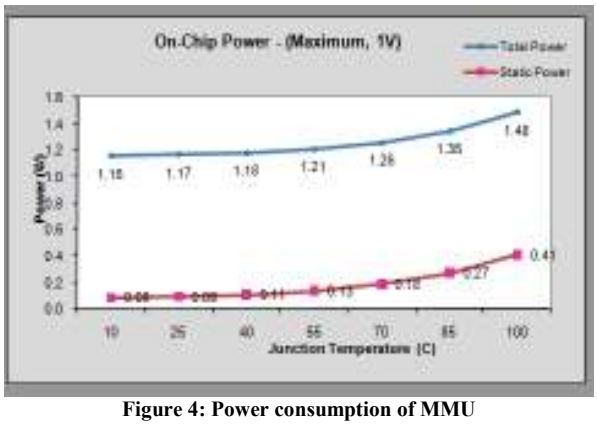



**Figure 4: Power consumption of MMU**

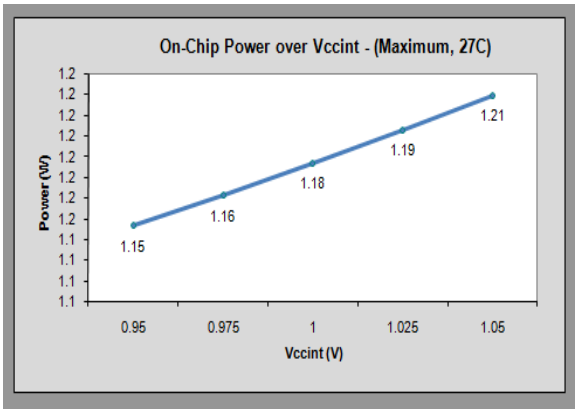



**Figure 5: On-chip power over internal Vcc**

## CONCLUSION

In this paper, the task allocation problem on the hybrid main memory composed of PRAM and DRAM is implemented. The energy efficiency of PRAM and the long write endurance of DRAM is exploited. The objectives as to minimize the energy consumption, number of writes on PRAM, and the PRAM size is achieved and the hybrid main memory system shows near optimal performance without page migration. The well-designed hybrid memory decreases energy delay product compared to DRAM-only system.

## REFERENCES


1. Barroso LA,Hölzle U. The case for energy-proportional computing, Computer. 2007; 40(12): 33.
2. Burr GW, Breitwisch MJ, Franceschini M, Garetto D, Gopalakrishnan K, Jackson B, Kurdi. B, Lam C, Lastras LA, Padilla A, Rajendran B, Raoux S, Shenoy RS.Phase change memory technology,J. Vac. Sci. Technol. B. 2010; 28(2): 223.
3. Dhiman G, Ayoub R,Rosing T. PDRAM: A hybrid PRAM and DRAM main memory system, in Proc. 46th Des. Autom. Conf. 2009; 664.
4. Dong X, Wu X, Sun G, Xie Y, Li H, Chen Y. Circuit and micro architecture evaluation of 3D stacking magnetic RAM (MRAM) as a universal memory replacement, in Proc. 45th Des. Autom. Conf. 2008; 554.
5. Garey MR, JohnDS. Computers and Intractability: A Guide to the Theory of NP-Completeness. New York: W. H. Freeman. 1979.
6. Hosom MI, Yamagishi H, Yamamoto T, Bessho K, Higo Y, Yamane K, Yamada H.
7. Shoji M, Hachino H, Fukumoto C, Nagao H, Kano H. A novel non-volatile memory with spin torque transfer magnetization switching: Spin-RAM," in Proc. IEEE IEDM. 2005;459.
8. Hu J, Xue C, Zhuge Q, Tseng W, Sha EHM. Toward energyefficient hybrid on-chip scratch pad memory with non-volatile memory, in Proc. Conf. Des., Autom. Test Eur. 2011; 1.
9. Lee BC, Ipek E, Mutlu O, Burger D. Architecting phase change memory as a scalable DRAM alternative, in Proc. 36th Int. Symp. Comput. Arch. 2009; 2.
10. Mogul JC, Argollo E, Shah M, Faraboschi P. Operating system support for NVM+DRAM hybrid main memory," in Proc. 12thWorkshop Hot Topics Operat. Syst. 2009; 14.
11. Qureshi MK, Srinivasan V, Rivers JA. Scalable high performance main memory system using phase-change memory technology,in Proc. 36th Int. Symp. Comput. Arch. 2009; 24.
12. Raoux S, Burr GW, Breitwisch MJ, Rettner CT, Chen YC, Shelby RM, Salinga M, Krebs D, Chen SH, Lung, Lam CH, Phase-change random access memory: A scalable technology,"IBM J. Res. Develop. 2008; 52(4): 465.
13. Shi L, XueC, Zhou X. Cooperating write buffer cache and virtual memory management for flash memory based systems," in Proc. IEEE Real-Time Embedded Technol. Appl. Symp. 2011; 147.
14. Seok H, Park Y, Park KH. Migration based page caching algorithm for a hybrid main memory of DRAM and PRAM, in Proc.26th ACM Symp. Appl. Comput. 2011; 595.
15. Tanzawa T, Tanaka T. A dynamic analysis of the dickson charge pump circuit," IEEE J. Solid-State Circuits. 1997; 32(8): 1231.
16. Xue CJ, Zhang Y, Chen Y, Sun G, Yang JJ, Li H. Emerging non-volatile memories: Opportunities and challenges, in Proc. 7th IEEE/ACM/IFIP Int. Conf. Hardw. /Softw. Codes. Syst. Synth. 2011; 325.


**Source of support: Nil, Conflict of interest: None Declared**