# Evaluation of algorithms reduce write energy in Phase Change Memories (PCM)

Hadi Mardani Kamali
90210294
hadi.mardani.kamali@gmail.com

Mahsa Rezaei Firuzkuhi
90702326
mahsa.rezaeifiruzkuhi@gmail.com

Supervisor:

Hossein Asadi
*Department of Computer Engineering*
*Sharif University of Technology*
asadi@sharif.edu

***Abstract***

Phase Change Random Access Memory (PCRAM or PCM) is one of the newest technologies using in memory systems and Storage devices. This class of memory is best candidate for replacing flash memories. It's concluded from some of considerable features such as **addressability**, which means the independence of PCM blocks, **scalability** which means the power of extension and developing alongside other technologies and **high capacity** such as SSDs.

In contrast these characteristics, there is a big problem in PCMs which that write energy consumptions is high. This circumstance causes there is a big challenge for using PCMs that it's desirable to use PCMs instead of Flash Memories and write energy consumption is a big knot which prevents that PCMs are introduced as a new generation of Main Memories.

There are some algorithms to reduce write energy consumption significantly. These methods basically proposed based on addressability and independence of PCM blocks. Some kind of methods improved write energy consumption by manipulation of data based on a specific algorithm and some kinds methods improved write energy consumption by using a placement algorithm based on free PCM blocks and using a specific signature.

This paper describes evaluation of proposed algorithm, their cons and pros and a hybrid algorithm that combined data manipulation and placement algorithms to reduce write energy consumption. Generally, this algorithm find the best free PCM block to overwrite new data which has minimum changes, but for choosing this free block it may to manipulate the original data(for example, in some cases writing invert of original data results better free block).we implement this algorithm and compare the result with proposed algorithm and show the hybrid algorithm can reduce write energy consumption.

## 1. *Introduction*

Phase Change Random Access Memories is one of the newest technologies in non-Volatile Memories which recently became the most important candidate for replacing flash memories [1]. As pointed out in [2, 3] scalability of recent NVMs like DRAMs is in jeopardy that may prevent DRAMs to scale beyond 30nm. But the structure of resistive memories based on arranging atoms within a cell and measuring resistive parameters of these cells are promising as a more scalable replacement for DRAMs and flash Memories [4].

There are some technologies have categorized in Resistive memories, like Spin-Torque-Transfer Magnetoresistive Random Access Memories (STT-MRAM), Ferroelectric Memories (FRAM) and PCM, but some considerable features of PCMs against other resistive Memories such as addressability, fast read access, better write endurance cause that in this category PCM is closest to commercialization [5].

Despite these features, there is a big problem in write operations. Because of the structure of building blocks in PCM we can't see good proportion and symmetry relative between read and write access energy. For read and write operations in PCM there are two different states which show different resistance for read and write access. These resistive parameters distinguish by driving of given magnitude and period current in PCM cells. The period of this process for write access against read access is very much and causes that the proportion of write energy consumption is more than read energy consumption about several orders of magnitude [6-8].

However, reducing write energy consumption primarily is based on the technology, nature and material of manufacturing of PCMs, but the architecture of using these memories can reduce write energy consumption. High write energy leads researchers to this idea that for reducing write energy consumption the simplest way is to reduce number of write operations. Because of addressability and independence of PCM cells, we can use various methods to reduce number of write operations [9, 10].

All efforts in proposed methods and algorithms in this issue is based on reducing the number of write operations in all application to saving wasted write energy. One type of these methods changes the content of data to reduce number of cells will be updated, for instance using inverting method is popular in this issue. These methods changes the content of data based on content aware operations and it's achieved by reading the content of data in all operations.

Other type of these methods uses a structure to select the best placement of data. This idea is achieved based on two points. First, independency of PCM cells and second, there are a lot of free PCM cells instantly. In fact, in these methods for each write operation based on a symbol which it's better to call signature shows the approximation of data and new data will be written to a free PCM cell that has the closest approximation data.

In this paper we investigate these methods and represent a hybrid algorithm based on placement algorithm and changing data before writing operations and show that this hybrid method reduces write energy consumption against cited method with a considerable proportion. As a general view, this hybrid method using placement algorithm, but for choosing best free PCM cell, not only the approximation of original data is compared with new data, but also the other type of changed data (for instance, invert of data) is compared and select the best approximation. Using some types of data such as inverting extend probability space of experiments and lead to better results.

The reminder of this paper is organized as follows. Section 2 reviews the basic knowledge about PCM storage technology. Section 3 describes manipulation technique for reducing number of write. Section 4 describes placement method for choosing best approximation. Section 5 shows our hybrid proposed algorithm. Section 6 describes the detail of additional functions and circuits and their overheads and disadvantages. Section 7 reviews the related and future works. Section 8 summarizes the paper.

## 2. *Background*

Phase change memory technology is based on two different electrical parameters. Each PCM cell consist of two basic properties. First, a Chalcogenide layer ($Ge_2Sb_2Te_5$) between top electrode and it's resistive and second, a resistive layer between Chalcogenide layer and bottom electrode. In each cell we can see a transistor (Bipolar or MOSFET) which control the access to cell. Fig.1 illustrates a general view of each PCM cell.

Steerage current with different magnitudes and durations on this element lead Chalcogenide to two states, "amorphous phase" which shows low resistance and determines the cell is RESET, "crystalline state" which shows high resistance and determines the cell is SET. Fig.2 shows the difference of these two states. The amorphous (RESET) is a immediately high magnitude pulse which occurs in a short time but has high magnitude, but crystalline (SET) is a smoothly low magnitude pulse which occurs in a long time but has low magnitude [11].

These operations need high energy pulse to program each cell but reading from cells is based on a low energy operation merely by sensing the resistance of cell. The structure of PCM cells and independency of cells shows that evaluation of cells against programming is less costly. Therefore using some methods that can reduce number of writes by using more read operations and improve total energy consumption especially in write operations.

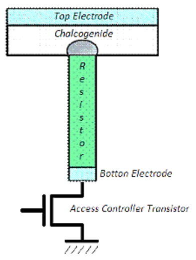


**Figure 1: Phase Change Random access Memory cell**

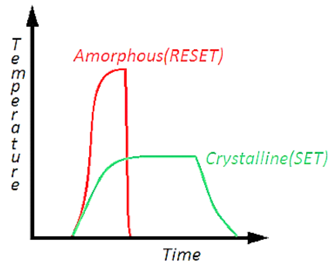


**Figure 2: Temperature of Programming in Chalcogenide**

## 3. *Manipulation Content Technique*

The general idea in manipulation content technique is based on evaluation of content of stored data before writing the new data. First intuitive idea is that to write just different cell. structure called selective Memory write and proposed based on independence of PCM cells. In this method, we read the cell which has been selected before writing new data and merely which cells that are different with old cells will been updated. Two points in this situation is important. First, it's important when this method is working. Note that because of the read energy consumption is much less than write energy consumption, so extra read operations are not critical in this case. Second, worse latency which is occurred for more read operations is not a new problem, because the read latency is much less than write latency. For instance, read time is about several tens ns but write time is about several hundred ns.

According to this issue that we use random I/O trace for evaluation of proposed method, so it's obvious that for random data the quantity of changed bits in average is half of all bits. So we can achieve up to 50 percent reduction in number of write operations.

Second intuitive method in this issue is one of the most important techniques in low power designs which called bus-inverting or XOR masked write method. In this method, before writing new data, the stored data read and compare with new data. If difference bits are more than half of data width then the inverse of original data is stored and if difference bits are less than half of data width then the original data is stored. One flag is used to represent that the original data is stored or inverse of original data. In comparison with Selective memory write, this method can reduce at least 50 percent of number of write.

As shown in table 1, it's obvious that for a sample write request bus inverting method has better result and can reduce number of write more than 50 percent. But in bus inverting method we use an overhead to show which type of data is using, for example if flag is equal with "0", it means that the original data has been written but if flag is equal with "1", it means that the inverse of data has been written. Another important issue is the function (circuit) that uses to determine the overhead and type of data.

**Table 1: writing new data example by using Selective Write and Bus Inverting**

| | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | flag | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| old data | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | ▨ | |
| new data | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | ▨ | Write Reduction (percentage) |
| selective write | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | ▨ | 31.25% |
| Bus Inverting | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 64.71% |

We can evaluate results in two ways. First, comparing number of write and ignoring detail of energy consumption. Second, we can evaluate based on energy consumption. First way is simpler but second way has more precisely results. In second way we need to simulate a PCM cell for evaluation of energy consumption for reads and writes (including sets and resets) and a gate (circuit) level design for computations of Bus Inverting flag and type of given data.

As figured out in [12] there are experimental results of simulation PCM cell in Verilog-A. We can use these results for scaling energy consumption of all PCM cells. Table 2 shows the major parameter for simulation PCM cell. With these parameters we can achieve reads and writes latency and energy consumption. Therefore, we can to scale the result for all PCM cells.

**Table 2: PCM cell parametric values**

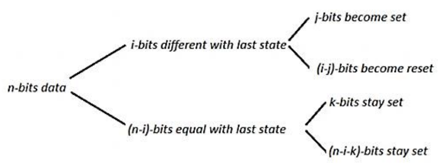
| Parameter | Value |
|---|---|
| Static resistance of Set (RSet) | 7KΩ |
| Programming current of Set(ISet) | 600µA |
| Programming time of Set (TSet) | 100ns |
| Static resistance of Reset (RResett) | 200KΩ |
| Programming current of Reset (IReset) | 1200µA |
| Programming time of Reset (TReset) | 100ns |
| Holding voltage (Vh) | 0.45V |
| Threshold voltage (Vth) | 0.78V |
| Dynamic-on resistance (Ron) | 1KΩ |

According to Table 2 and based on other simulation on PCM cell we can approximately estimate read, write (including set and reset) energy consumption approximately $E_r\colon E_{set}\colon E_{reset} = 0.2\colon 1\colon 5$. These values normalized based on $E_{set}$ which is about 50pj [7]. Also, for estimation of bus inverting flag energy consumption we can consider based on a gate level design which has been evaluated by power compiler and this implementation shows for each word size the energy consumption is half of word size per "pj". In fact, if the word size is "n" bit then bus inverting flag energy consumption is "n/2 pj".

Following computation shows energy consumption of three methods include direct method (without comparing), selective memory write and bus inverting method:

- $\mathrm{E}_{\text{Non-Select Or XOR Masked}} = \frac{n}{2}(\mathrm{E}_{\text{set}} + \mathrm{E}_{\text{reset}})$
- $\mathrm{E}_{\text{Selective Mem Write}} = \frac{n}{4}(\mathrm{E}_{\text{set}} + \mathrm{E}_{\text{reset}}) + n * \mathrm{E}_{\text{read}}$

There are four major conditions for n-bit data in XOR masked write:



Based on above tree we should compute four state:

$$\mathrm{E}_{0\to 0} \xLeftrightarrow{\substack{\text{(Number of diff} \\ \text{less than} \\ \text{number of equality)}}} j\mathrm{E}_{\text{set}} + (i - j)\mathrm{E}_{\text{reset}}$$

$$\mathrm{E}_{0\to 1} \xLeftrightarrow{\substack{\text{(Number of diff} \\ \text{more than} \\ \text{number of equality)}}} (n - i - k + 1)\mathrm{E}_{\text{set}} + k\mathrm{E}_{\text{reset}}$$

$$\mathrm{E}_{\widehat{d_p}=0}(i,j,k) = \min\left(\mathrm{E}_{0\to 0}(i,j,k), \mathrm{E}_{0\to 1}(i,j,k)\right)$$

Like above computation we have the same manner for $\widehat{d_p}$=1. Therefore we can calculate total energy consumption.

- $\mathrm{E}_{\text{bus-inverting}} = \sum_{i=0}^{n}\sum_{j=0}^{i}\sum_{k=0}^{n-i} \mathrm{P}_r(i,j,k)(\mathrm{E}_{\widehat{d_p}=0} + \mathrm{E}_{\widehat{d_p}=1}) + \mathrm{E}_{\text{Search}} + n.\mathrm{E}_{\text{Read}}$

$$\mathrm{P}_r(i,j,k) = \binom{n}{i}\binom{i}{j}\binom{n-i}{k}\left(\frac{1}{2^n}\right)$$

Note that above equation represent number of probability of occurrence of changes. So with above estimations and calculations we can evaluate manipulation content technique and compare energy consumption with energy consumption without any technique. Fig.3 shows that with Selective memory write method we achieve to approximately %40 in write energy consumption and in bus inverting method the maximum efficiency is %55. In Bus Inverting method it's obvious that extending word size reduce efficiency of write energy consumption. The reason of this reduction is that by extending word size the binomial distribution become normal distribution. In fact, by increasing word size Hamming Distance is increased and using bus inverting method is not efficient. The energy consumption percentages are achieved by comparing with normal operations with direct writing without any evaluation data. Note that overhead energy consumption involves in bus inverting flag energy consumption.

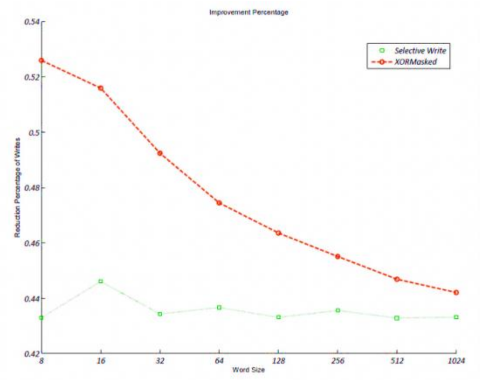


**Figure 3: Write energy saving results in Data Manipulation technique**

## 4. *Content based placement Technique*

General idea in this method is based on implementation a placement algorithm to select a good free block for minimizing the number of changes between last stored and new data. This method needs reading some characteristic of stored data before writing new data and compare with characteristic parameters of new data. This condition needs an overhead for each block that called signature like bus inverting method. Signature is an overhead for each PCM cell that represent that the format of stored data. In fact, if two data have equal signature then those data have approximately equal data and the changed cells are minimized.

This method uses one of important algorithm that called "Data Comparison Write". DCW is an algorithm for optimal writing based on minimizing number of writes. According to independency of writes in individual of PCRAM Cells this method have a desirable result and can considerably reduce number of writes in this type of memories [10].

However, using an intelligent algorithm for optimal placement is desirable but the cost of this process is high. Searching algorithm for finding best free block for placement and computing circuit for signature is critical issue in this section. But implementation of this method shows that there is one order of magnitude improvement in realistic application.

The general idea is being formed of two subsections. One section is computing signature for each block and another section is block placement decision.

In this method signatures are based on number of ones. However increasing signature size led us to better result and more precisely signature but the format of signature is fixed-length and small as possible. We use some "sets" which shows the resolution of precision, but there is one big problem for few sets and decreasing number of sets has no desirable results. If we divide block to n-sets with same size and maximum of signature width is m-bit signature we have "mn"-bits for signature of block. Therefore more sets and signature width cause more precise but there is a trade-off between word size of signature as an overhead and precise.

For deciding of block placement suppose that PRAM controller has this ability to distinguish free blocks. Format of addressability is based on an index that for a given signature the index mapped to one list that all members has same signature. For new data after computing signature two states is possible.

Note that in this method we assume there is an intelligent mechanism operated by Operating system which determines which blocks are free.

One state is that there is at least one free block with same signature. This method uses a variable to limit the number of same signature free blocks. This variable is search distance limit and this limitation is known as a tool for result that can support various conditions. For example increasing this variable can have better result but the overhead of computation is high. On the other hand the major goal of definition this limitation is that it is probably there are several free blocks with same signature with given new data.

Second state is that there is no free block with same signature. This method just for first state uses an intelligently way to minimize the number of write and for second state write the new data in head of free blocks. This decision is one of criticizing point of proposed method, while that we can use a method to find closest free block.

We need to evaluate our method on a standard trace data. In last method that called Bus Inverting method we use random data, but in this situation random data is one of worst case for using DCW. Fig.4 and fig.5 illustrate the distribution of data in random data and TIF image data. These figures drew by using two "sets" of data. Based on these figures it's obvious that for random data we have dense distribution in half of changed bit for each "set", but in TIF image data, distribution of data for using this algorithm is more prefer. In next section the hybrid method is based on Bus Inverting and content aware placement, therefore for evaluation of data between these methods we use random data.

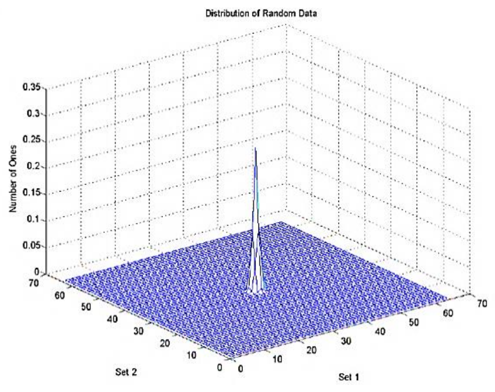


**Figure 4: Distribution of Ones in request data block of random Trace**

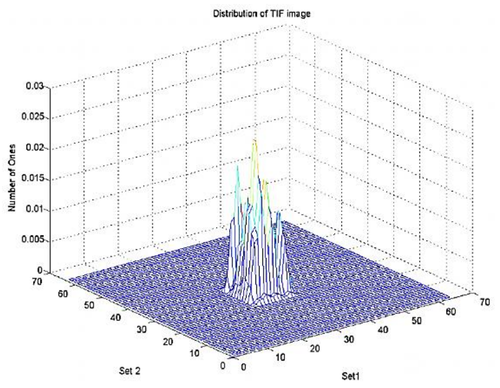


**Figure 5: Distribution of Ones in request data block of TIF Trace**

Fig.6 shows Suspend-to-Disk request trace based on checking some office documents by user. Using of our DCW method for this trace has an effective result which shows the content aware algorithm has considerable saving energy for some practical fields.

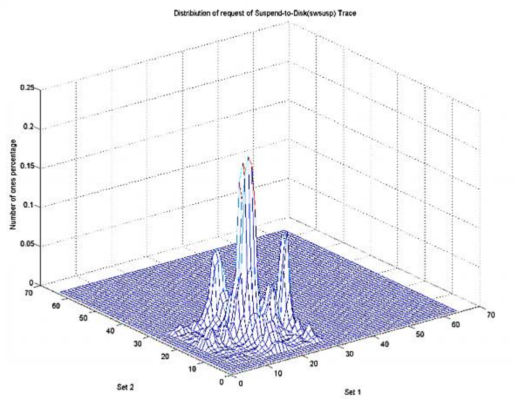


**Figure 6: Distribution of Ones in request data block of swsusp Trace**

Fig.7 shows the reduction of number of writes by using this method. Note that in this figure the value of set is fixed and is equal with 2.

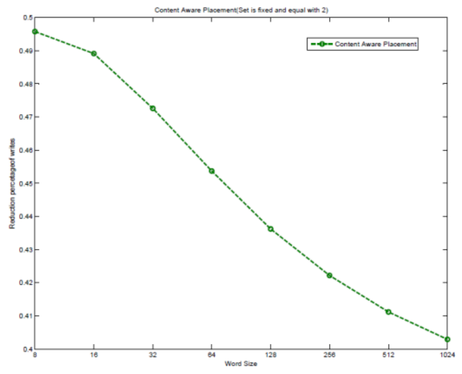


**Figure 7: Write energy saving results in Content aware Placement for random request trace (fixed set)**

Fig.8 shows the reduction of number of writes by using this method by using swsusp data block request. Note that in this figure the value of set is fixed and is equal with 2. In comparison with random request trace, we achieved to about %15 and in comparison with direct method (without any technique) we achieved to %65 improvement.

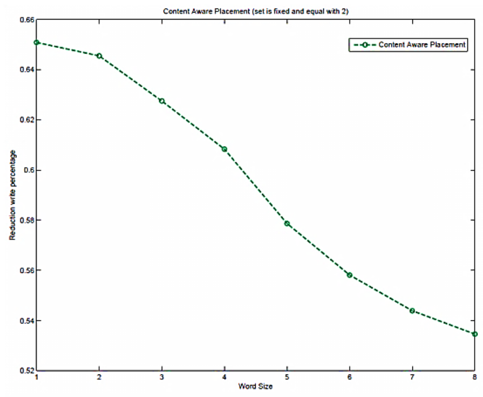


**Figure 8: Write energy saving results in Content aware Placement for swsusp request trace (fixed set)**

An intuitive perception shows that for random data block trace data changing number of sets and bits per set have no considerable effect on reduction of number of writes. Fig.9 illustrates this issue that increasing number of sets have a little reduction in number of writes.

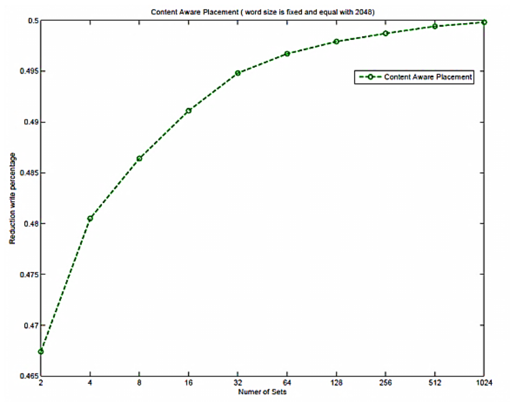


**Figure 9: Write energy saving results in Content Placement for random request trace (fixed word size)**

For representation new hybrid method we need some type of trace data which has been evaluated in above techniques. Although the result of random data is not suitable and the maximum reduction is about %60 we'll evaluate new hybrid method by using random trace data to improve achieved results by means of above techniques.

## 5. *Hybrid Proposed Method*

Bus inverting method reduced number of writes by manipulation of content of data without any changing in placement of data. Content aware placement method reduces number of writes by using a content aware placement algorithm without any change in content of data. Each method has an advantage which is a big disadvantage in another method.

In Bus inverting method we ignored all free block which it may help us to improve reduction of writes. Also, in content aware placement method we can use manipulation content of data alongside the placement algorithm for more efficiency. In fact, this is an intuitive perception that combination of these methods can give better result.

There are some disadvantages in bus inverting and content aware placement method which will be covered by combination of these methods. As mentioned in bus inverting method, one of the biggest problems is ignoring the free blocks which can use for writing. Although using these free blocks needs to have a mapping algorithm, but the operating system is thanks to this issue. Therefore we can use a placement algorithm to achieve better results. In other hand, using bus inverting method in content aware placement method has this feature that we have bigger search space for selecting best free block for writing new data. Suppose that we don't have any free block with same signature with new data. If we use invert of original data the probability of this inequality is lower than first state and the search space is bigger.

General idea is simple and is combination of bus inverting and content aware method. In this method, when a new request is coming, the signature of new data is computed and the signature of new data is compared with signature of blocks. Afterward, the invert data signature will be compared with signature of free blocks. In this situation, we select free block which is more similar with new data and signature of new data. So, we can achieve more reduction in number of writes against bus inverting method and content aware placement.

Fig.10 shows that the difference of reduction between content aware method and hybrid algorithm. This figures illustrates that hybrid algorithm has considerable reduction against content aware placement method for lower word size. As mentioned in section 3 and fig.3 increasing word size reduce the efficiency of bus inverting method, so in this figure we can see that we have same manner and by increasing word size the efficiency of hybrid algorithm will tend to content aware placement algorithm. Therefore, hybrid algorithm has better result for lower word size.

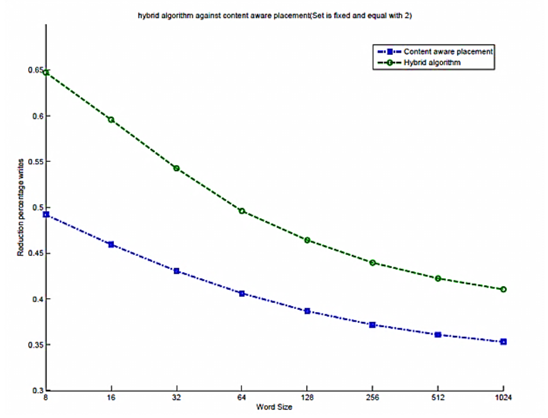


**Figure 10: Write energy saving results in Content aware Placement and hybrid algorithm (fixed sets)**

## 6. *Evaluation of Extra functions energy dissipation*

As mentioned in section 3, we have two ways to evaluate and compare efficiency of these algorithms. First, comparing number of writes and ignoring detail of energy dissipation. Second, we can evaluate based on energy consumption with more details. But the important issue is that we should achieve to better result in two ways. Therefore we have to evaluate additional functions (Circuits) which used for reduction energy consumption.

Based on experimental results which figured out in [7, 9] we can approximate energy consumption order for these additional circuits. Table 3 shows an experimental result which prove that the energy of additional functions is not considerable against write energy consumption. It's because of the simplicity of additional functions. In each model the algorithm works based on comparison and comparison functions is based on XOR circuit. Therefore the complexity of additional functions is merely based XOR circuit. However, XOR energy consumption is not negligible but against the quantity of write energy consumption in PCM cell is not considerable. So, we can use first way to evaluate results and compare methods based on number of write operations.

**Table 3: comparison function energy dissipation and latency**

| word size | Energy Consumption (pJ) | latency (ns) |
|---|---|---|
| 8 | 4.5 | 4.5 |
| 16 | 9.7 | 5.2 |
| 32 | 17.5 | 6.1 |
| 64 | 34 | 6.9 |
| 128 | 63 | 7.5 |
| 256 | 120 | 8.2 |
| 512 | 249 | 8.9 |
| 1024 | 497 | 9.6 |

## 7. *Related and Future Works*

One of the most important issues which represented here is using some type of coding and information theory algorithms by using some especial technique to have better resolution of data with lower bit width. In information theory and coding, there are various coding algorithm may have better efficiency. However, finding effective coding method is a little complex but this achievement leads us to more and considerable efficiency.

Another issue is useful to consider is compression of data. There are some compression methods which work based on difference of stream of data. These methods are not only simple to implement but also lead us to more saving energy dissipation. First compression method which is using in this situation is Pulse Coding Modulation (PCM). PCM data are transmitted as a serial bit stream of binary-coded time-division multiplexed words that works based on difference of two consecutive data. Because of dependency of stream data, the difference of consecutive data is little and it's occurred in all situations. Therefore, we can use lower word size for coding data and achieve to more saving write energy consumption.

There are some discussable coding methods which work based on level changing of data that can lead us to better result like binary bit representations includes NRZ-X, Biphase-X. these methods work based on changing level of data.

## 8. *Conclusion*

This paper presented some techniques to reduce write energy consumption. These techniques worked based on two important points about PCRAM cells. First, independency and addressability in PCRAMs caused and second, free PCM blocks. Based on manipulation content technique which writes new data based on last data and number of changes and content aware placement technique which writes new data based on free blocks and similarity of computed signature of new data and free blocks we presented a hybrid method which use two above techniques. In fact, in this method, choosing free block for writing new data is not only based on original data, but also is dependent to inverse of original data. The results shows that in hybrid method we achieved to about %65 overall improvement and about %15 improvement against content aware placement technique.

## 9. *References*

[1] F. Bedeschi et al., "A multi-level-cell bipolar-selected phase-change memory," in Proceedings of the IEEE International Solid-State Circuits Conference, Feb. 2008, pp. 428–625.

[2] K. Kim. "Technology for sub-50nm DRAM and NAND ash manufacturing". International Electron Devices Meeting, 2005.

[3] André B. Bondi, 'Characteristics of scalability and their impact on performance', Proceedings of the 2nd international workshop on Software and performance, Ottawa, Ontario, Canada, 2000, pages 195 – 203

[4] G. Dhiman, R. Ayoub, and T. Rosing. "PDRAM: A hybrid PRAM and DRAM main memory system". Proceedings of 47th Design Automation Conference, June 2009.

[5] G.W. Burr et al., "Overview of candidate device technologies for storage-class memory," IBM Journal of Research and Development, vol. 52, no. 4/5, pp. 449–464, July/Sept. 2008.

[6] H. Oh et al., "Enhanced Write Performance of a 64-Mb Phase-Change Random Access Memory," Journal of Solid-State Circiuts, vol. 41, no. 1, pp. 122–126, Jan. 2006.

[7] S. Kang et al., "A 0.1-μm 1.8-V 256-Mb Phase-Change Random Access Memory (PRAM) with 66-MHz Synchronous Burst-Read Operation," Journal of Solid-State Circuits, vol. 42, no. 1, pp. 210–218, Jan. 2007.

[8] T. Nirschl et al., "Write strategies for 2 and 4-bit multi-level phase-change memory," in Proceedings of the IEEE International Electron Devices Meeting, Dec. 2007, pp. 461–464.

[9] W. Xu, J. Liu, T. Xhang "Data Manipulation Techniques to Reduce Phase Change Memory write Energy," International Symposium on Low Power Electronics and Design, Aug. 2009.

[10] B. Wongchaowart, M. K. Iskander, S.cho "A Content-Aware Block Placement Algorithm for Reducing PRAM Storage Bit Write," Mass Storage Systems and technologies (MSST),2010 IEEE 26th symposium, May. 2010.

[11] S. Raoux et al., "Phase-change random access memory:A scalable technology," IBM Journal of Research and
Development, vol. 52, no. 4/5, pp. 465–479, 2008.

[12] Y. Liao, Y. Chen, M.Chiang "Phase-change memory modeling using Verilog-A," Behavioral Modeling and Simulation Workshop, 2007. BMAS 2007. IEEE International , Vol.47, pp. 159–164, Sept. 2007.

## 10. *Appendix*

### 10.1. *Manipulation content technique*

1-1. Evaluation of bit writes without any methodology (SLC)

```
function Energy=NoMeth(NumReq,Size)

    data = int16(rand(2^16,Size)); %64K*Size random 
data
    %EnergyRead = 0.2;
    EnergySet = 1;
    EnergyReset = 5;

    TotalSet = 0;

    TotalReset = 0;
    EnergyTotal = 0;


    %Numberof1 = sum(sum(data));
    %Numberof0 = 2^16*Size-Numberof1;

    for i=1:NumReq
        Addr = int32(rand*2^16);
        %OldData = data(Addr,1:Size);
        NewData = int16(rand(1,Size));
        data(Addr,1:Size) = NewData;
        NumSet = sum(NewData);
        NumReset = Size-NumSet;
        TotalSet = TotalSet+NumSet;
        TotalReset = TotalReset+NumReset;
        %EnergyTotal = EnergyTotal +

      ((NumSet*EnergySet)+(NumReset*EnergyReset));

    end
     Energy = TotalSet*EnergySet +
TotalReset*EnergyReset;

end
```

1-2. Evaluation of bit writes with selective memory write (SLC)

```
function Energy=Selective(NumReq,Size)

    data = int16(rand(2^16,Size)); %64K*Size random
data

    EnergyRead = 0.2;
    EnergySet = 1;
    EnergyReset = 5;

    TotalSet = 0;
    TotalReset = 0;
    TotalRead = 0;

    for i=1:NumReq
        Addr = int32(rand*2^16);
        OldData = data(Addr,1:Size);
        NewData = int16(rand(1,Size));
        for j=1:Size
            if (NewData(j) ~= OldData(j))
                data(Addr,j) = NewData(j);
                TotalRead = TotalRead+1;
                if(NewData(j) == 1)
                    TotalSet = TotalSet+1;
                else
                    TotalReset = TotalReset+1;
                end
            else
                data(Addr,j) = OldData(j);
                TotalRead = TotalRead+1;
            end
        end

        %EnergyTotal = EnergyTotal +
        ((NumSet*EnergySet)+(NumReset*EnergyReset));
    end

    Energy = TotalSet*EnergySet +
TotalReset*EnergyReset +
                TotalRead*EnergyRead;
end
```

1-3. Evaluation of bit writes with Bus Inverting method (SLC)

```
function Energy=XORMasked(NumReq,Size)


    data = int16(rand(2^16,Size)); %64K*Size random
data
    data(1:2^16,Size+1) = 0; %invert bit

    EnergySearch =Size/100;
    EnergyRead = 0.2;
    EnergySet = 1;
    EnergyReset = 5;

    TotalSet = 0;
    TotalReset = 0;
    TotalRead = 0;
    TotalSearch = 0;


    for i=1:NumReq
        Addr = int32(rand*2^16);
        OldData = data(Addr,1:Size+1);
        NewData = int16(rand(1,Size));
        NewData(Size+1) = 0;
        HammDis = xor(NewData , OldData);
        if (sum(HammDis)>=Size/2)
            WrittenData = not(NewData);
            TotalSearch = TotalSearch+1;
        else
            WrittenData = NewData;
            TotalSearch = TotalSearch+1;
        end
        for j=1:Size+1
            if (WrittenData(j) ~= OldData(j))
                data(Addr,j) = WrittenData(j);
                TotalRead = TotalRead+1;
                if(WrittenData(j) == 1)
                    TotalSet = TotalSet+1;
                else
                    TotalReset = TotalReset+1;
                end
            else
                data(Addr,j) = OldData(j);
                TotalRead = TotalRead+1;
            end
        end
        %EnergyTotal = EnergyTotal +

((NumSet*EnergySet)+(NumReset*EnergyReset));
    end

    Energy = TotalSet*EnergySet +
TotalReset*EnergyReset +
                TotalRead*EnergyRead +
TotalSearch*EnergySearch;
end
```

## *10.2. Content Aware Placement technique*

2-1. Distribution of random trace data by using 2 sets

```
function testDist()

     for i=1:20000
     data=int16(rand(2^9,8));

     set1(i)=int16(sum(sum(data(1:2^8,1:8)))/(2^8*8)*2^6
);

     set2(i)=int16(sum(sum(data(2^8+1:2^9,1:8)))/(2^8*8)
*2^6);
     end
     for i=1:64
     pst1(i)=(20000-nnz(set1-i))/20000;
     pst2(i)=(20000-nnz(set2-i))/20000;
     end

     st1=1:64;
     st2=1:64;
     prob=transpose(pst1)*pst2;
     mesh(st1,st2,prob);
end
```

2-2. Distribution of TIF trace data by using 2 sets

```
function testDistImg()

     data=image2arr();

     for i=1:128
     temp=data((i-1)*2^9+1:i*2^9,1:8);

     set1(i)=int16(sum(sum(temp(1:2^8,1:8)))/(2^8*8)*2^6
);

     set2(i)=int16(sum(sum(temp(2^8+1:2^9,1:8)))/(2^8*8)
*2^6);
     end
     for i=1:64
     pst1(i)=(128-nnz(set1-i))/128;
     pst2(i)=(128-nnz(set2-i))/128;
     end

     st1=1:64;
     st2=1:64;
```

```
    prob=transpose(pst1)*pst2;
    mesh(st1,st2,prob);
end
```

2-3. converting image tow array binary data

```
function data=image2arr()

    image = imread('cameraman.tif');
    for i=1:length(image)
   for j=1:length(image)
       data((i-
1)*length(image)+j,1:8)=de2bi(image(i,j),8);
   end
    end

end
```

2-4. signature computation algorithm

```
function data=SignBit(data,bitwidth,set)

    data(1,bitwidth+1) = 0; % block is free or not free
    for j=1:set %set declare product of set and bit.
                if(sum(data(1,1+(j-

1)*bitwidth/set:j*bitwidth/set))>bitwidth/(2*set))
                    data(1,bitwidth+1+j) = 1;
                else
                    data(1,bitwidth+1+j) = 0;
                end
    end

end
```

2-5. Placement method

```
function NumWR=placement(size,bitwidth,set,req)

    data = int16(rand(size,bitwidth));

    for i=1:size
   data(i,1:bitwidth+set+1) =
SignBit(data(i,1:bitwidth),bitwidth,set);
   data(i,bitwidth+set+2) = i;
    end



    NumWR = 0;

    for iter=1:req
   newdata =
SignBit(int16(rand(1,bitwidth)),bitwidth,set);
   newdata(1,bitwidth+1) = 1;
   newdata(1,bitwidth+set+2) = 0;


   for i=1:size
         HammDiss(i,1:bitwidth+2) =
[xor(newdata(1,1:bitwidth),data(i,1:bitwidth))

sum(xor(newdata(1,1:bitwidth),data(i,1:bitwidth))) i];
   end

   SortDist = sortrows(HammDiss,(bitwidth+1));

   index = 1;
   if ~isequal(data(1:size,bitwidth+1),ones(size,1)) %we don't
have any free cell
       while
data(SortDist(index,bitwidth+2),bitwidth+1)==1
           index = index+1;
       end
       chngcell =

sum(xor(data(SortDist(index,bitwidth+2),1:bitwidth+1+set),newdata(1,1:bitwidth+1+se
t)));
         data(SortDist(index,bitwidth+2),1:bitwidth+set+1)=
newdata(1,1:bitwidth+set+1);
         data(SortDist(index,bitwidth+2),bitwidth+set+2)

=SortDist(index,bitwidth+2);
       NumWR = NumWR+chngcell/(bitwidth+1+set);

   else
       data(size+1,1:bitwidth+set+1) =
newdata(1,1:bitwidth+set+1);
       data(size+1,bitwidth+set+2) = size+1;
       size = size+1;
       NumWR = NumWR+1+(set+1)/(bitwidth+set+1);
   end

    end

    NumWR = NumWR/req;
end
```

## 10.3. *Hybrid Algorithm*

3-1. Hybrid algorithm

```
function NumWR=hybrid(size,bitwidth,set,req)

    data = int16(rand(size,bitwidth));

    for i=1:size
        data(i,1:bitwidth+set+1) =
SignBit(data(i,1:bitwidth),bitwidth,set);
        data(i,bitwidth+set+2) = i;
        data(i,bitwidth+set+3) = 0; %invert bit
    end

    NumWR = 0;
    NumWR2 = 0;
    for iter=1:req

   newdata =
SignBit(int16(rand(1,bitwidth)),bitwidth,set);
   newdata(1,bitwidth+1) = 1;
   newdata(1,bitwidth+set+2) = 0;
   newdata(1,bitwidth+set+3) = 0;

   for i=1:size
        HammDiss(i,1:bitwidth+2) =
[xor(newdata(1,1:bitwidth),data(i,1:bitwidth))

sum(xor(newdata(1,1:bitwidth),data(i,1:bitwidth))) i];
   end

   for i=size+1:2*size
        HammDiss(i,1:bitwidth+3) =
[not(xor(newdata(1,1:bitwidth),data(i-
                                                size,1:bitwidth)))
            sum(not(xor(newdata(1,1:bitwidth),data(i-
size,1:bitwidth)))) i-size 1];
   end

   SortDist = sortrows(HammDiss,(bitwidth+1));

   index = 1;
   if ~isequal(data(1:size,bitwidth+1),ones(size,1))
%we don't have any free cell
       while
data(SortDist(index,bitwidth+2),bitwidth+1)==1
           index = index+1;
       end
       if SortDist(index,bitwidth+3)==1
           newdata(1,1:bitwidth) =
not(newdata(1,1:bitwidth));
           newdata(1,bitwidth+2:bitwidth+set+3) =

not(newdata(1,bitwidth+2:bitwidth+set+3));
       end

       chngcell =

sum(xor(data(SortDist(index,bitwidth+2),1:bitwidth+1+set),n
ewdata(1,1:bitwidth+1+set)));

data(SortDist(index,bitwidth+2),1:bitwidth+set+1) =

newdata(1,1:bitwidth+set+1);
       data(SortDist(index,bitwidth+2),bitwidth+set+2)
= SortDist(index,bitwidth+2);
       data(SortDist(index,bitwidth+2),bitwidth+set+3)
= SortDist(index,bitwidth+3);
       NumWR = NumWR+chngcell/(bitwidth+1+set);

   else
```

```
        data(size+1,1:bitwidth+set+1) = newdata(1,1:bitwidth+set+1);
        data(size+1,bitwidth+set+2) = size+1;
        size = size+1;
        NumWR = NumWR+1;
    end

     end

     NumWR = NumWR/req;
end
```