ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

Design and Analysis of Low Power Error Tolerant Cache Memory Using Dynamic Multistep Tag Comparison

Abilash .V[1], Chidambaram. S[2]

PG Scholar, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India [1]

Associate Professor, Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India [2]

ABSTRACT: Tag bits in cache memories are also exposed to transient errors but a few efforts have been made to reduce their vulnerability. The method was to exploit prevalent same tag bits to improve error protection capability of the tag bits in the caches using Same Tag Information (STI). When data are fetched from the main memory, it is checked if adjacent cache lines have the same tag bits as those of the data fetched. The modification done by using Counting Bloom Filter (CBF) scheme, which makes output only with comparatively reduced power. The estimated area and delay is not much reduced.A partial tag enhanced Bloom Filter (pBF) to improve the accuracy of the cache miss prediction method and hot/cold checks that control data liveness to reduce the tag comparisons of the cache hit prediction method. Both methods are also combined so that their order of application can be dynamically adjusted to adapt to changing cache access behavior, which further reduces tag comparisons. Experimental results showed that the proposed method reduces the energy consumption of tag comparison by an average of 50%, area used is reduced by an average of 71% and delay is reduced by 53.35% when compared with existing methods.

KEYWORD*S*: Cache, Tag Information, Counting Bloom Filter, Hit, Miss, Data Liveness **.**

I. INTRODUCTION

Cache memory is usually part of the central processing unit, or part of a complex that includes the CPU and an adjacent chipset where memory is used to hold data and instructions that are most frequently accessed by an executing program - usually from RAM-based memory locations. CPU cache memory operates between 10 to 100 times faster than RAM, requiring only a few nanoseconds to respond to the CPU request. RAM cache, of course, is much speedier in its response time than magnetic media, which delivers I/O at rates measured in milliseconds. A CPU cache places a small amount of memory directly on the CPU. This memory is much faster than the system RAM because it operates at the CPU's speed rather than the system bus speed. The idea behind the cache is that chip makers assume that if data has been requested once, there's a good chance it will be requested again. Placing the data on the cache makes it accessible faster. Same tag bits are used to improve error protection capability of the tag bits in the caches. When an error is detected in the tag bits, the same tag bit information is used to recover from the error in the tag bits.

II. SAME TAG INFORMATION

Exploiting prevalent same tag bits to improve error protection capability of the tag bits in the caches. When data are fetched from the main memory, it is checked if adjacent cache lines have the same tag bits as those of the data fetched. This same tag bit information is stored in the caches as extra bits to be used later. When an error is detected in the tag bits, the same tag bit information is used to recover from the error in the tag bits. Before accessing data from main memory to CPU it is stored in the cache memory for the processing waiting as required. During the storage in the cache memory the data can be misplaced in the corresponding address. This misplacement is known as Transient Error.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

III. COUNTING BLOOM FILTER

A Counting Bloom filter is a space-efficient probabilistic data structure, conceived by Burton Howard Bloom in 1970, that is used to test whether an element is a member of a set. False positive matches are possible, but false negatives are not, thus a Bloom filter has a 100% recall rate. In other words, a query returns either "possibly in set" or "definitely not in set". Elements can be added to the set, but not removed (though this can be addressed with a "counting" filter). The more elements that are added to the set, the larger the probability of false positives. Bloom proposed the technique for applications where the amount of source data would require an impractically large amount of memory if "conventional" error-free hashing techniques were applied. He gave the example of a hyphenation algorithm for a dictionary of 500,000 words, out of which 90% follow simple hyphenation rules, but the remaining 10% require expensive disk accesses to retrieve specific hyphenation patterns. With sufficient core memory, an error-free hash could be used to eliminate all unnecessary disk accesses; on the other hand, with limited core memory, Bloom's technique uses a smaller hash area but still eliminates most unnecessary accesses.

IV. HASH FUNCTION STRUCTURE

An example of a Bloom filter, representing the set { x, y, z }. The top arrows show the positions in the bit array that each set element is mapped to. The element wise not in the set { x, y, z }, because it hashes to one bit-array position containing 0. For this figure, m = 18 and k = 3. An empty Bloom filter is a bit array of m bits, all set to 0. There must also be k different hash functions defined, each of which maps or hashes some set element to one of the m array positions with a uniform random distribution. To add an element, feed it to each of the k hash functions to get k array positions. Set the bits at all these positions to 1. To query for an element (test whether it is in the set), feed it to each of the k hash functions to get k array positions. If any of the bits at these positions is 0, the element is definitely not in the set – if it were, then all the bits would have been set to 1 when it was inserted. If all are 1, then either the element is in the set, or the bits have by chance been set to 1 during the insertion of other elements, resulting in a false positive. In a simple Bloom filter, there is no way to distinguish between the two cases, but more advanced techniques can address this problem. Removing an element from this simple Bloom filter is impossible because false negatives are not permitted. An element maps to k bits, and although setting any one of those k bits to zero suffices to remove the element, it also results in removing any other elements that happen to map onto that bit. Since there is no way of determining whether any other elements have been added that affect the bits for an element to be removed, clearing any of the bits would introduce the possibility for false negatives. One-time removal of an element from a Bloom filter can be simulated by having a second Bloom filter that contains items that have been removed. However, false positives in the second filter become false negatives in the composite filter, which may be undesirable. In this approach re-adding a previously removed item is not possible, as one would have to remove it from the "removed" filter. It is often the case that all the keys are available but are expensive to enumerate (for example, requiring many disk reads). When the false positive rate gets too high, the filter can be regenerated; this should be a relatively rare event. -

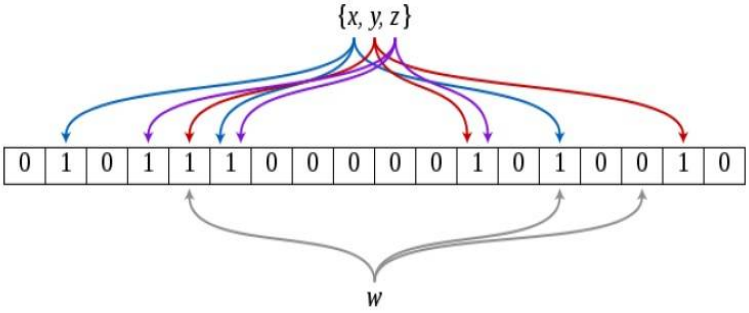

Fig.1 Hash Function Structure

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

V. PARTIAL TAG BLOOM FILTER (PBF)

In a Counting Bloom filter, the component 'Prob' will mention as "True Positive", "False Negative" and "False Positive". In which true positive means data is present and false negative means data is not present.
"False Positive" is the multi detected address denotation, which simply means "data can be considered. The probability of the occurrence of a false
positive need to be reduced to increase the energy gain enabled by the Bloom filter. In this work, it is aimed to reduce the probability of a false positive by equipping the Bloom filter with partial tags.

VI. TAG BIT ERRORS

Introduced Tag Bit Similarity for error correction in cache memory for reducing transient error. A standard taxonomy for error detection classifies errors as undetected, detected but uncorrectable (DUE), or detected and correctable errors. Single event upsets in logic path can potentially cause Silent Data Corruptions (SDCs) in unmitigated system. SDCs are undetected errors that lead to incorrect machine results. DUE occurs when detected errors exceed error protection capability. Tag arrays are typically protected.

VII. PSEUDO HIT

A pseudo-hit refers to a hit that is actually a miss in the absence of a transient error. Fig. 2 shows an example of a pseudo-hit. The pseudo-hit may lead the data path to use the incorrectly matched data. However, pseudo-hit can be detected by storing check bits for each row of tag bits. When an access for a given input results in a hit, the corresponding check bits of the tag bits are read out and then compared with the check bits computed from input tag bits. Any mismatch denotes that the tag bits have been corrupted and the hit is surely a pseudo-hit.

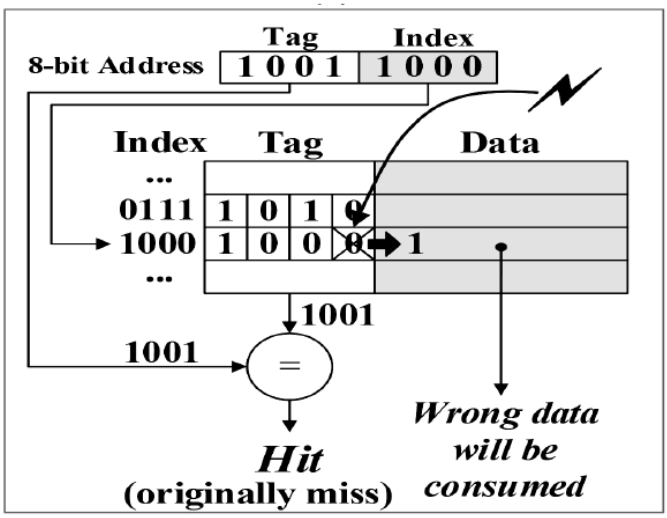

Fig.2 Example of Pseudo Hit

VIII. PSEUDO MISS

A pseudo-miss refers to a miss that is actually a hit when there is no transient error. Fig. 3 shows an example of a pseudo-miss. Pseudo-misses may cause a data integrity problem in a write-back data cache, which holds dirty data. A pseudo-miss on dirty data triggers a fetch and use of stale data from a lower level. In case of a write-through cache, however, a pseudo-miss does not cause a problem. It just incurs a cache miss

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

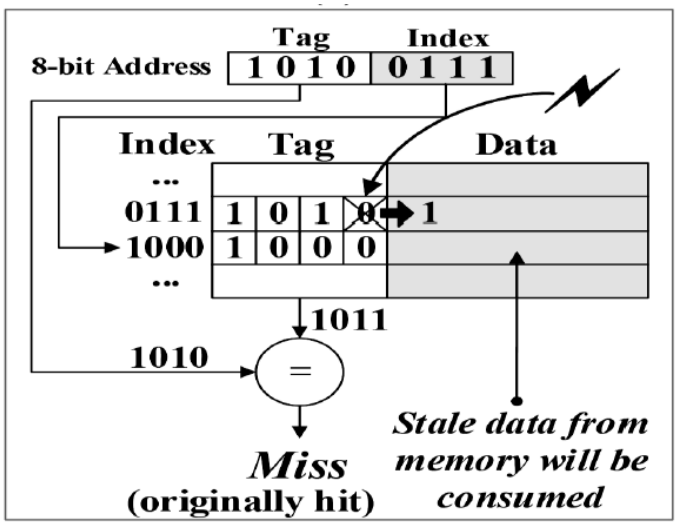

Fig.3 Example of Pseudo Miss

IX. MULTI HIT

In addition, transient errors may result in multi-hit errors when caches are associative. Fig.4 shows an example of a multi-hit in a four-way set associative cache. At most one tag bit must be matched in one cache set, but it is possible that multiple tag bits are matched in a single set due to a pseudo-hit. Multi-hit errors occur infrequently, but their rate can be increased in highly associative caches.

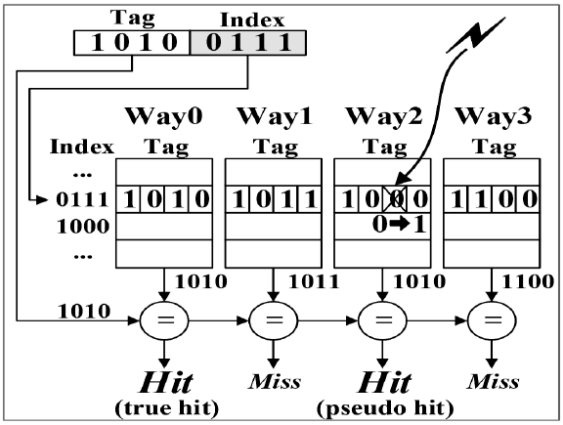

Fig.4 Example of Multi Hit

X. MULTISTEP TAG COMPARISON APPROACH

In this proposed method, tag comparison in a highly associative cache consumes a significant portion of the cache energy. In existing methods for tag comparison reduction are based on predicting either cache hits or cache misses. A novel ideas for both cache hit and miss predictions are obtained.

This method of dynamic multistep tag comparison combines both cache hit prediction (hot and cold cache line checks) and cache miss prediction (partial tag enhanced Bloom filter) methods. A partial tag enhanced Bloom

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

filter (pBF) is presented to improve the accuracy of the cache miss prediction method and hot/cold checks that control data liveliness to reduce the tag comparisons of the cache hit prediction method. We also combine both methods so that their order of application can be dynamically adjusted to adapt to changing cache access behavior, which further reduces tag comparisons. To overcome the common limitation of multistep tag comparison methods, it is proposed by a method that reduces tag comparisons while meeting the given performance bound. It is dynamic because it dynamically adjusts the order of tag comparison steps to maximize the efficiency of each of the cache hit and miss prediction methods.

XI. RESULTS AND DISCUSSIONS

The simulation result of L2 cache with counting Bloom filter method and dynamic multistep tag comparison method is obtained with XILINX ISE 14.1 using VHDL language. The area, delay and power consumed have been reported.The output for existing CBF shown in the Fig.5, Hot Line and Cold Line Cache enhanced methods are shown in the Fig.6 and Fig.7 respectively.

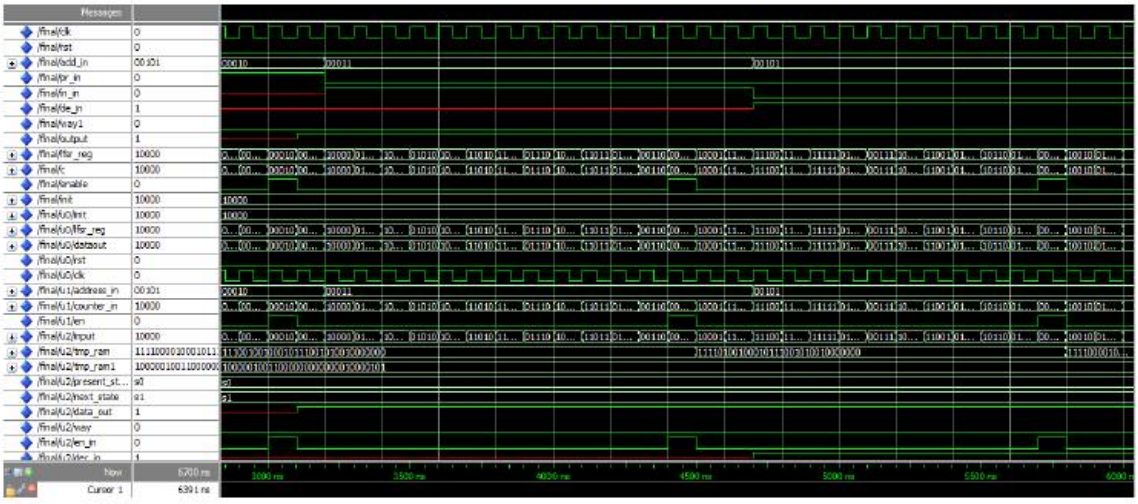

Fig.5 Existing CBF Method

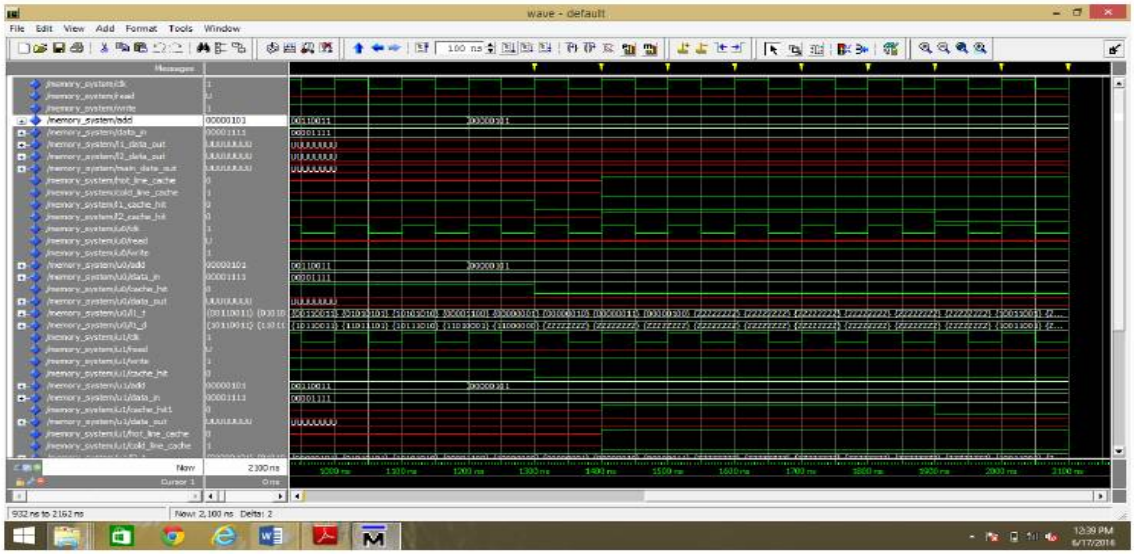

Fig.6 Hot Line Cache enhanced

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 3, March 2017

5th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '17)

13th-14th March 2017

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India.

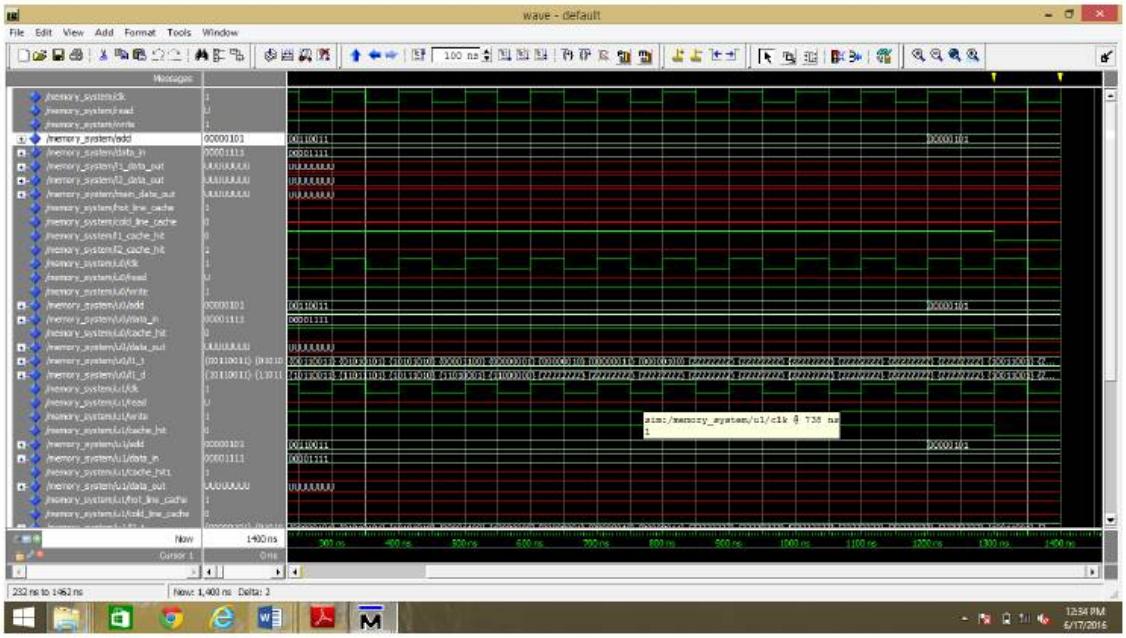

Fig.7 Cold Line Cache enhanced

XII. CONCLUSION

The dynamic multistep tag comparison method to reduce the energy consumed in tag comparison within highly associative L2 caches is proposed. It is a combination of partial tag-enhanced Bloom filter to improve the accuracy of cache miss prediction and hot/cold checks (with dynamic timeout tracking) as a cache hit prediction method. In future, we will investigate the effectiveness of the proposed method in multicore environments such as 3-D graphics-based user interfaces and high-end embedded systems such as smartphones and tablet PCs.

REFERENCES

[1] Hyunsun Park, Sungjoo Yoo and Sunggu Lee, "A Multistep Tag Comparison Method for a Low-Power L2 Cache," IEEE Transactions on Computer-Aided Design Of Integrated Circuits and Systems, Vol. 31, No. 4, April 2015.

[2] Jeongkyu Hong, Jesung Kim, and Soontae Kim, "Exploiting Same Tag Bits to Improve the Reliability of the Cache Memories," IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 23, No. 2, February 2015.

[3] Jianwei Dai and Lei Wang, "An Energy-Efficient L2 Cache Architecture Using Way Tag Information Under Write-Through Policy," IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 21, No. 1, January 2013.

[4] Soontae Kim, "Reducing Area Overhead for Error-Protecting Large L2/L3 Caches," IEEE Transactions On Computers, Vol. 58, No. 3, March 2009.

[5] Koustav Bhattacharya, Nagarajan Ranganathan and Soontae Kim, "A Framework for Correction of Multi-Bit Soft Errors in L2 Caches Based on Redundancy," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 17, No. 2, February 2009.

[6] Zhigang Hu, Stefanos Kaxiras and Margaret Martonosi, "Timekeeping in the Memory System: Predicting and Optimizing Memory Behavior," IEEE Transactions on Dependable and Secure Computing, Vol. 3, No. 3, July- September 2006.

[7] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen and Norman P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," IEEE Transactions On Computers, Vol. 54, No. 12, December 2005.

[8] Wei Zhang, "Replication Cache: A Small Fully Associative Cache to Improve Data Cache Reliability," IEEE Transactions On Computers, Vol. 54, No. 12, November 2005.

[9] Patrick J. Meaney, Scott B. S., Pia N. Sanda, and Lisa Spainhower, "IBM z990 Soft Error Detection and Recovery" IEEE Transactions on Device and Materials Reliability, Vol. 5, No. 3, September 2005.

[10] Charles W. Slayman, "Cache and Memory Error Detection, Correction, and Reduction Techniques for Terrestrial Servers and Workstations" IEEE Transactions on Device and Materials Reliability, Vol. 5, No. 3, September 2005.