L2 Cache Architecture Using Wag Tag Information

S.Vishnupriya[1]

PG Student., M.E(Vlsi Design).,K.P.R Institute Of Engineering And Technology,Coimbatore,Tamil Nadu,India

***Abstract*— Cache write through policy is being employed in many high performance microprocessor in order to improve the performance and in achieving good tolerence to the soft errors in on-chip caches.The most critical issue in cache design is power dissipation. Way-tagged cache architecture is used to improve the energy efficiency of write through caches.The way tags of L2 cache are maintained in L1 cache during read operation and in this proposed technique the L2 cache works in a direct-mapping manner during write hit.In this technique a tag is being attached to each way in the L2 cache.The data is loaded from the L2 cache to L1 cache.By utilizing the way tags stored in way tag array the L2 cache can be accessed as a direct mapping cache during write hits so the cache energy consumption is being reduced.This way tagging idea can be applied to the existing low power cache techniques.This technique can be implemented along with the filters so that the power consumption can be further reduced.**

***Index Terms* —Cache, way tagged cache, cache hit and miss,power consumed.**

I INTRODUCTION

Cache lines is that transfer of data between memory and cache in blocks of fixed size.A cache entry is being created when a cache line is copied from memory into cache.The cache entry includes the copied data and the requested memory location.When a read or write operation has to take place in the main memory the corresponding entry in the cache is being checked.The contents of the requested memory location is being checked by the cache in any cache line that may contain the required address.A cache hit has occurred when the processor finds the memory location is in the cache. If the memory location has not occurred then a cache miss has occurred.

Processor immediately reads or writes the data in the cache line is known as cache hit.A cache miss is when the cache allocates a new entry and the data is copied from main memory.When the number of requests that can be served from cache increases then the overall system performance becomes faster.Caches are relatively small to be cost efficient and to use the data efficiently.If the data is requested again that has been recently used then the references exhibit temporal locality. If the data is requested that is physically store close to data that has been requested already then references exhibit spatial locality.

II RELATED WORK

The partitioned cache data arrays into sub banks was proposed by Suetal.,(1997).Ghose et al.,(1999) proposed the way of dividing the cache bit lines into small segmentations.Way concatenation is the another technique proposed by Zhang et al.,(2003) which reduces the cache energy in embedded systems.Another approach is to employ the redunant cache to predict the incoming references was proposed by Min et al.,(2004).The location cache needs to be triggered for each and every operation in L1 cache which wastes energy if the hit rate is higher in L1 cache. When comparing the above related work phase caches and way predicting caches are most commonly used in case of high performance processors. This proposed technique achieves better efficiency with no performance degradation.The basic idea is to keep the smaller number of most recently used addressess.For phased caches energy consumption of accessing tag array accounts for a significant portion of total L2 cache energy.

III CACHE ARCITECTURE

Here a conventional set associative cache system is considered where the L1 data cache loads or writes the data from or into the L2 cache.The L2 caches are activated simultaneously for the performance consideration. Fig 3.1 illustrates the architecture of two-level cache.IN case of write through policy the L2 cache maintains the most recent copy of the data.So whenever a data is being updated in the L1

cache then the L2 cache is also being updated with the same data as well.The result of this is increase in the write accesses to the L2 cache.

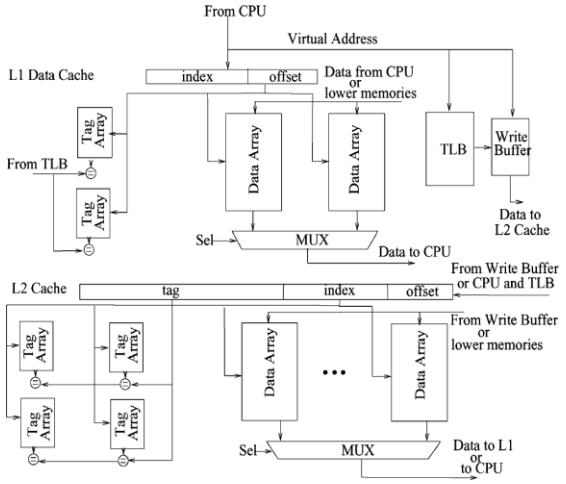


Fig1: Conventional cache architecture

A. Two-level cache

The locations of the data copies in L1 cache will not change till the data are evicted from the L2 cache.The proposed cache exploits the fact to reduce the number of ways to access during L2 cache.When the L1 data cache loads the data from L2 cache then the way tag of the data in the L2 cache is also sent to the L1 cache and it is being stored in a new set of way-tag arrays.

B.Read and Write in Cache

Generally both write and read accesses in the L1 cache may need to access the L2 cache.In case of proposed cache system different operations are being accessed.In case of write through policy all the write operations of the L1 cache need to access the L2 cache.When write hit occurs in L1 cache only one way in L2 cache wil be activated because way tag information is available from the way tag arrays we can obtain the L2 way of accessed data.For write miss in the L1 cache the required data is not stored in the L1 cache.All ways of L2 cache have to be activated in same time.In order to avoid performance degradation way tag arrays has to be accessed at the same time. For read operations in case of l1 cache neither read hits nor misses need to access the way-tag arrays because read hits do not need to access the L2 cache,while in read misses the corresponding way tag information is not available in the way-tag arrays.

IV WAY TAGGED CACHE

A.Way-tag arrays

In the proposed cache each cache line in the L1 cache keeps its L2 way tag information in the corresponding entry of the way tag arrays as shown in figure 4.1, where only one L1 data array and associated way-tag array are shown.When a data is loaded into L1 cache from L1 cache, the way tag of the data is written into the way-tag array.When the data is being updated in the L1 data cache the corresponding data in L2 cache needs to be updated as well.The way tag is stored in the way tag array is then forwarded to the way-tag buffer along with the

data from L1 data cache.The data arrays and the way tag arrays share the same address because the mapping between the two is exclusive.When the read and write signal of the way tag arrays is generated from the read or write signal of the data arrays in the L1 data cache as shown in the fig.4.1. A control signal known as UPDATE is got from cache controller.During the read operation of the L1 cache the way tag arrays need not be accessed and so they are deactivated to reduce energy overhead.

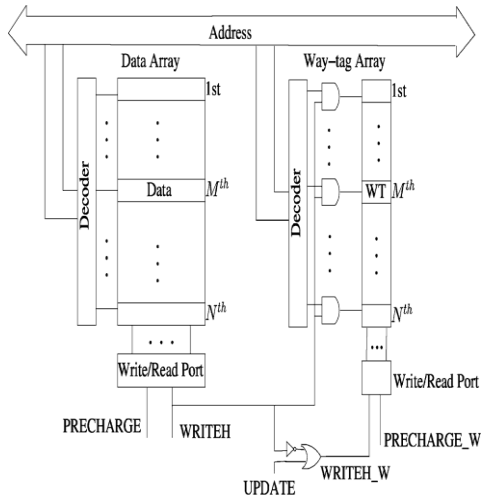


Fig2: Way tag arrays

B. Way tag buffer

This temporarily stores the way tag reads from the way tag array.The implementation of the way tag buffer is shown in fig.4.2. It has the same number of

entries as the write buffer of L2 cache and the control signal is shared with it.Each entry of the way tag buffer has n+1 bits.where n is the line size of way tag arrays.An additional ststus bit is there which shows whether the operation is in the current entry is a write miss on the L1 data cahe.All the ways of L2 cache need to be activated when a write miss occurs as the way information is not available. It is updated with the read operations of way tag arrays at the same clock cycle.

C. Way decoder

Way decoders function is to decode way tags and activate the desired ways present in the L2 cache.The line size of the way tag array is n=log2 N bits,where N is the number of ways in the l2 cache.So the energy is being minimized.For a L2 write access the way decoder works as n-to-N decoder that selects one way enable signal.This operates along with the decoders of the tag and data

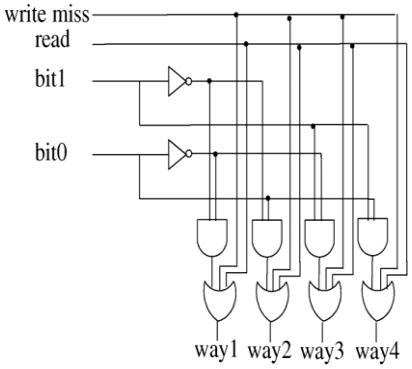


Fig3:Way decoder

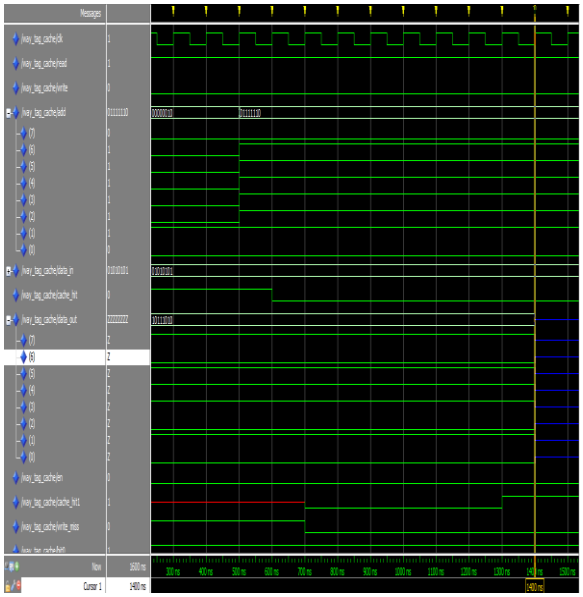
arrays in the l2 cache.In case of a write miss or a read miss in L1 cache so that all ways in L2 cache are activated.

D. *Way register*

Way register provides the way tags for the way tag array.in case of four way L2 cache labels “00”,”01”,”10”,”11” are stored in the way register and tagging in the L2 cache and the corresponding way tag is sent to way tag arrays.this operates under different modes during different kinds of operation.

VI. RESULTS AND DISCUSSIONS

The results are being simulated for both the conventional and the existing architecture.In case of two level conventional cache architecture the cache hit and cache miss is being done in both the levels of the cache.Here both the data read and write operations takes places.The way tag cache is used to hit the data so that the power is consumed less than the conventional type. The power comparison of both the type is given and there is a great difference beween the two types.



VII. CONCLUSION

This paper presents a new energy efficient cache technique for high performance microprocessors employing the write-through policy.The proposed technique attaches a tag to each way in the L2 cache. This way tag is being sent to the way-tag arrays in the L1 cache when the data is loaded from the L2 cache to the L1 cache. Utilizing the way tags stored in the way-tag arrays, the L2 cache can be accessed as a direct-mapping cache during the subsequent write hits, thereby reducing cache energy consumption. The future work is to use the partial way tag in the way tag array to improve the accuracy of cache miss and to reduce the tag comparisons of cache hit.This reduces tag comparison and so the power is consumed.

REFERENCES

1. Ghose.K and Kamble .M. B, "Reducing power in superscalar processor caches using subbanking, multiple line buffers and bit-line segmentation," in Proc. Int. Symp. Low Power Electron. Design, 1999,

2. Lyon.T, Delano.E, McNairy.C, and Mulla.D, "Data cache design considerations for Itanium 2 processor," in Proc. IEEE Int. Conf. Comput. Design, 2002, pp. 356–362

3. Maiz.J, hareland.S, Zhang.K, and Armstrong.P, "Characterization of multi-bit soft error events in advanced SRAMs," in Proc. Int. Electron Devices Meeting, 2003, pp. 21.4.1–21.4.4.

4. Rusu.S, Stinson. J, Tam. S, Leung. J, Muljono .H, and Cherkauer.B, "A 1.5-GHz 130-nm itanium 2 processor with 6-MB on-die L3 cache," IEEE J. Solid-State Circuits, vol. 38, no. 11, pp. 1887–1895, Nov. 2003.

5. Vera. X, Abella .J Gonzalez.A, and Ronen.R, "Reducing soft error vulnerability of data caches," presented at the Workshop System Ef- fects Logic Soft Errors, Austin, TX, 2007.

6. Mitchell.J, Henderson.D, and Ahrens.G, "IBM POWER5 processor based servers: A highly available design for business-critical applications," IBM, Armonk, NY, White Paper, 2005.

7. Inoue.K, Ishihara.T, and Murakami.K, "Way-predicting set-associative cache for high performance and low energy consumption," in Proc. Int. Symp. Low Power Electron. Design, 1999, pp. 273–275.

8. Hasegawa.A, Kawasaki.I, Yamada.K, Yoshioka.S, Kawasaki.S, and Biswas.P, "Sh3: High code density, low power," IEEE Micro, vol. 15, no. 6, pp. 11–19, Dec. 1995.

9. F. X. Ruckerbauer and G. Georgakos, "Soft error rates in 65 nm SRAMs: Analysis of new phenomena," in *Proc. IEEE Int. On-Line Test. Symp.*, 2007.

10. B. Brock and M. Exerman, "Cache Latencies of the PowerPC MPC7451," Freescale Semiconductor, Austin, TX, 2006.

11. J. Maiz, S. hareland, K. Zhang, and P. Armstrong, "Characterization of multi-bit soft error events in advanced SRAMs," in *Proc. Int. Electron Devices Meeting*, 2003.

12. A. Malik, B. Moyer, and D. Cermak, "A low power unified cache architecture providing power and performance flexibility," in *Proc. Int. Symp. Low Power Electron. Design*, 2000.

