DESIGN OF LOW POWER L1 CACHE FOR EMBEDDED PROCESSOR

T.Nathiya,
P.G.Scholar, Applied Electronics,
Shree Venkateshwara Hi Tech Engineering College,
Gobi, Tamil Nadu, India.

Mr.R.Kandasamy,M.E.,
Professor, Department of ECE
Shree Venkateshwara Hi Tech Engineering College
Gobi, Tamil Nadu, India.

***Abstract*—This report implements a novel energy-efficient cache architecture based on a matching mechanism that uses a reduced number of tag bits. The proposed architecture is based on moving a large subset of the tag bits from the cache into an external register (called the *Tag Overflow Buffer*) that serves as an identifier of the current locality of the memory references. Another technique also used that is Early Tag Access (ETA) to improve energy efficiency and to find destination way. It, thus, enables only the destination way to be accessed if a hit occurs during the ETA. This ETA cache can be configured under two operation modes to exploit the tradeoffs between energy efficiency and performance. It is shown that our technique is very effective in reducing the number of ways accessed during cache accesses. This enables significant energy reduction with negligible performance overheads. So the dynamic and static energy will reduce and improve energy efficiency.**

Key words: Early Tag Access, Tag Overflow Buffer.

I.INTRODUCTION

In this project concentrate to reduce l1 data cache energy or power and improve energy efficiency to maintain the better performance. REDUCING power consumption in cache memory is a critical problem for embedded processors that target low power applications. It was reported that on-chip. caches could consume as much as 40% of the total chip power Furthermore, large power dissipation could cause other issues, such as thermal effects and reliability degradation. This problem is compounded by the fact that data caches are usually performance critical. Therefore, it is of great importance to reduce cache energy consumption while minimizing the impact on processor performance.
Techniques:

- Way tagging techniques
- Way predicting techniques
- Way halting techniques

These techniques are used to reduce the cache energy. Here going to use early tag access technique and tag overflow buffering for l1data cache memory energy reduction. These are reduced power more than other techniques.
A level 1 cache (L1 cache) is a memory cache that is directly built into the microprocessor, which is used for storing the microprocessor’s recently accessed information, thus it is also called the primary cache. It is also referred to as the internal cache or system cache.

L1 cache is the fastest cache memory, since it is already built within the chip with a zero wait-state interface, making it the most expensive cache among the CPU caches. However, it has limited size. It is used to store data that was accessed by the processor recently, critical files that need to be executed immediately and it is the first cache to be accessed and processed when the processor itself performs a computer instruction. In more recent microprocessors, the L1 cache is divided equally into two: a cache that is used to keep program data and another cache that is used to keep instructions for the microprocessor. Some older microprocessors, on the other hand, make use of the undivided L1 cache and use it to store both program data and microprocessor instructions.

The early tag access (ETA) cache is improve the energy efficiency of data caches in embedded processor and it is determine the destination ways of memory instructions before the actual cache accesses. It, thus, enables only the destination way to be accessed if a hit occurs during the ETA. This energy-efficient cache architecture (TOB) based on a partial-tag scheme, which relies on the idea of bringing most of the tag bits outside the cache into a register that identifies the current locality. On a memory access, this register is first checked against the most significant bits of the address to determine whether we are inside the current locality or not. On a hit, the partial-tag cache is accessed normally (yet with a smaller cost); on a miss, the normal miss procedure is followed, with minor modifications.

The following are three of the main strengths of this scheme.

1) **It uses a fixed number of tag bits in the cache**. Therefore, it does not require any special activation circuitry to dynamically change the number of active tag bits, thus allowing to use a regular, yet smaller cache.
2) **It achieves a tag energy reduction comparable to other schemes, yet with a much smaller hardware overhead** (namely, one register, two comparators, and a 3 bit-counter in the most complex configuration). 3) **It achieves a reduction of static (i.e., leakage) energy**, as a side effect, thanks to the reduction of the overall number of bits in the tag arrays.

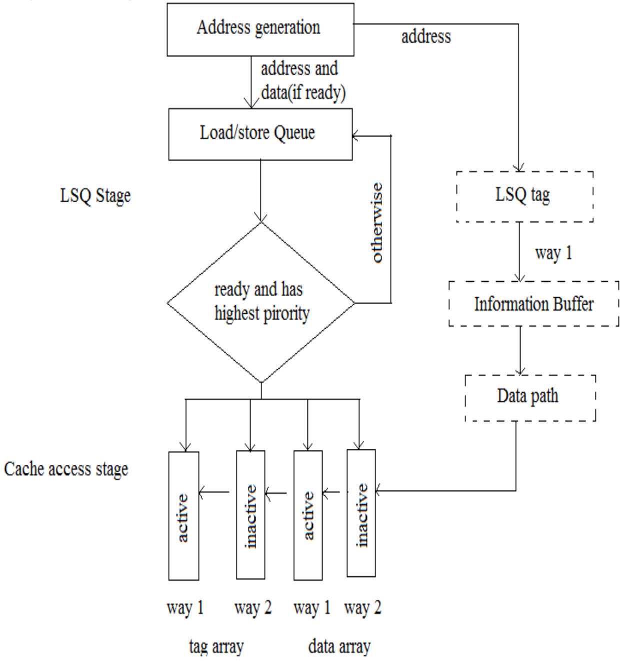

Fig.1.flowchart of L1 cache using ETA

II.PREVIOUS WORK

The Early Tag Access cache reduces the number of unnecessary way accesses thereby reducing cache energy consumption. To accommodate different energy and performance requirements in embedded processors. The ETA cache, each time a memory instruction is sent into the LSQ, an access to a new set of tag arrays and is performed, as shown in the dotted lines in Fig. 1.This new set of LSQ tag arrays are implemented as a copy of the tag cache needs to be activated and thus enables energy savings. The early destination way of a memory instruction determined at the LSQ stage is not the same as the actual one determined at the cache access stage due to the cache misses that happen during the LSQ lookup operation, the matched way in the LSQ tag arrays will be used as the destination way of this instruction when it accesses the L1 data cache subsequently. If this destination way is correct, only one way accessed in L1 data cache. instruction simply follows its early destination way, to access the L1 data cache.

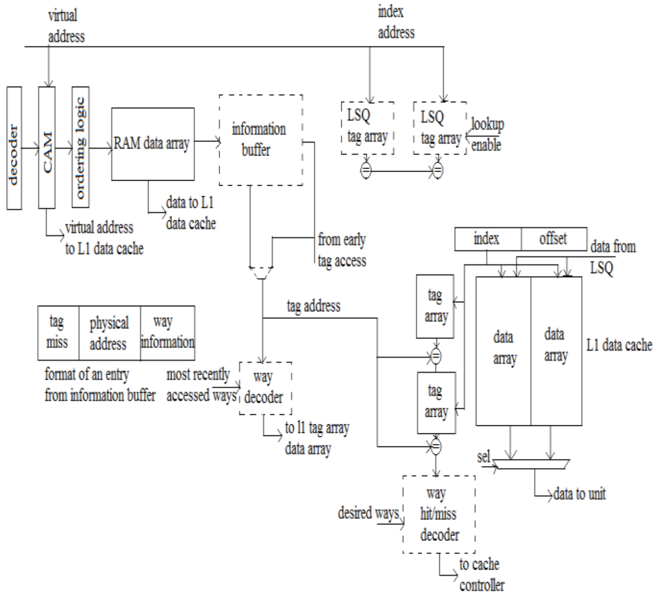

Fig.2.Block diagram of ETA cache

To avoid the data contention with the L1 data cache, the LSQ tag arrays and LSQ TLB are implemented as a copy of the tag arrays and TLB of the L1 data cache, respectively. There are two types of operations in the LSQ tag arrays and LSQ TLB: lookup and update. ETA cache block diagram are consists of

- LSQ Tag
- Information Buffer
- Way Hit/Miss Decoder
- Way Decoder

III. PROPOSED METHODOLOGY

A.REDUCED TAG CACHE ARCHITECHTURE

The TOB and the cache are always accessed in parallel at each memory reference. The outcome of the TOB lookup tells whether an access to the reduced-tag cache is feasible or not. The architecture works as follows. On a memory reference whether an access to the reduced-tag cache is feasible or not. The architecture works as follows. On a memory reference the most significant bits of the address are fed to the TOB. If the two values match (a *TOB hit* occurs), then we can safely access the reduced-tag cache without worrying about false hits. Clearly, the lookup in the cache may result in a hit or miss as any other access to a regular cache. A key issue for the efficient operations of this scheme is the choice of the value of the TOB. The most reasonable choice, at a slight expense of hardware complexity, is to allow the TOB value to change so as to dynamically adapt to possible changes in locality.

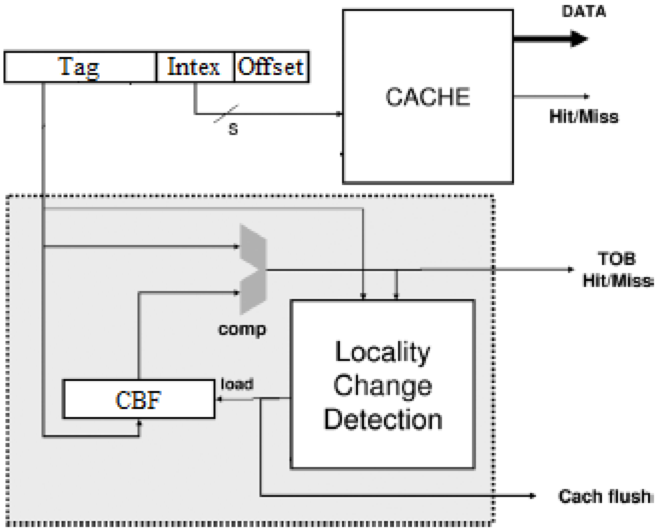

Fig.3.Architecture of Tag Overflow Buffer

This is the task of the block labeled in figure as “Locality Change Detection”, which, based on the observation of both the address and TOB miss output decides whether or not to enable the loading of a new locality value (i.e., the MSB bits of the current address). When this happens, the cache must be flushed, since all the values it contains actually refer to the previous locality.

B.COUNTING BLOOM FILTER

An increasing number of architectural techniques have relied on hardware counting bloom filters (CBFs) toimprove upon the energy, delay, and complexity of various processor structures. An area characteristics of two implementations for CBFs using full custom layouts in a commercial 0.13- m fabrication technology. . L-CBF compared to S-CBF is 3.7 or faster and requires 2.3 or 1.4 less energy depending on the operation. Checking whether a memory block is currently cached is an example of a member- ship test in processors. The CBF provides a definite answer. L-CBF accepts three inputs and produces a single-bit output is-zero.

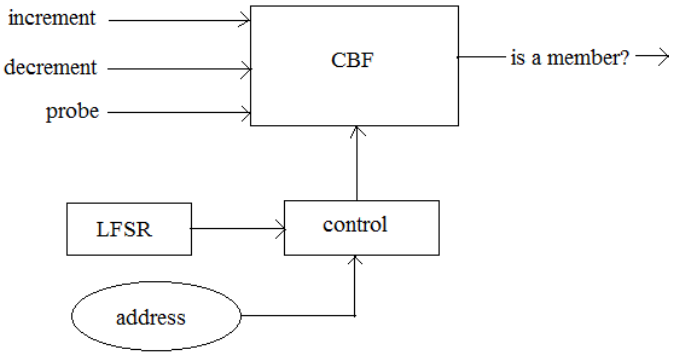

Fig.4.Block diagram of Counting Bloom Filter

The input operation select specifies the type of operation:

- Increment count (INC)
- Decrement count (DEC)
- Test if the count is zero (PROBE).

The first two operations increment or decrement the corresponding count by one, third one checks if the count is zero and returns true or false (single-bit output). We will refer to the first two operations as *updates* and to the third one as a *probe*.

C.LINEAR FEEDBACK SHIFT REGISTER (LFSR)

A linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state.

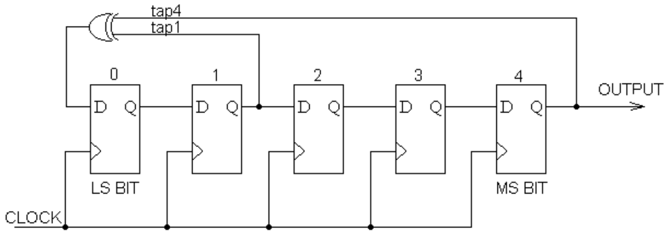

Fig.5.Circuit diagram of LFSR

The most commonly used linear function of single bits is exclusive-or (XOR). Thus, an LFSR is most often a shift register whose input bit is driven by the XOR of some bits of the overall shift register value. The initial value of the LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits which appears random and which has a very long cycle. This LFSR is used to give control signal to CBF for select member or not.

D.FLOWCHART OF L1 CACHE USING ETA AND TOB

ISSN: 2278 – 1323

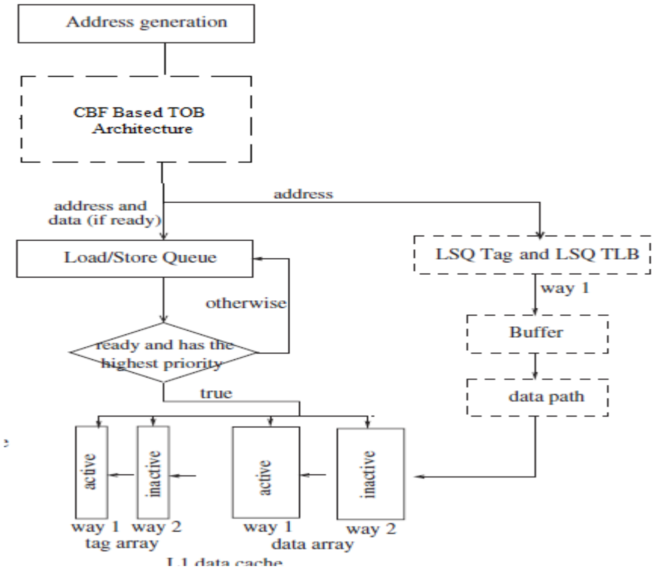

Fig.6.flowchart of L1 cache using ETA and TOB

Now see L1 cache flow chart using two techniques.

First the address is generated and it is goes to TOB architecture. Then this architecture is using matching mechanism for partial tag comparison. So here take most significant bits of tag address for comparison. Counting Bloom Filter is used to check that address is in or not and The LFSR is give signal to control that is enable or disable. For example address is there means the signal is enable and the cache is hit otherwise cache miss. The Cache hit mean further action is performed that is Early Tag Access. This Early Tag Access cache reduces the number of unnecessary way accesses thereby reducing cache energy consumption. Two types of operations are performed in LSQ tag.

- Lookup (read)
- Update (write)

This information stored in information buffer until next instruction will issue. It is keep that instruction in a period of one clock cycle. Then the desired way only active in L1 data cache stage using this technique. Suppose miss means to perform re-access.

IV.RESULT AND DISCUSSION

The previous method is using only ETA technique and it is energy savings ranging up to 52% of total energy. This method is using full tag address for comparison. But the proposed method is using partial tag address for comparison. So that proposed method is additionally added to previous method. Then here now using two techniques. There are Early Tag Access and Tag Overflow Buffering.

A.TABLE

	Ext(**Phase I**)	Pro(**Phase II**)
Time(**ns**)	7.505	6.420
Power(**mW**)	147	133

The above tabulation is show that power and time difference in between existing and proposed method. Time denoted the ns and power denoted as mw. The existing work is have 147 mw Power and 7.505 ns time.

V.CONCLUSION

This paper presented a new energy-efficient cache design techniques for low-power embedded processors. The proposed technique predicts the destination way of a memory instruction at the early LSQ stage with partial tag comparison. Here only one way needed to be accessed during the cache access stage if the prediction is correct, thereby reducing the energy consumption significantly and improve performance. future work is being directed toward extending this technique to other levels of the cache hierarchy and to deal with multithreaded workloads.

VI.REFERENCES

1. A. Ma, M. Zhang, and K. Asanovi(2001), "Way memoization to reduce fetch energy in instruction caches," in Proc. Int. Symp. Comput. Archit.Workshop Complex. Effect. Design, pp. 1–9.
2. C. Zhang, F. Vahid, Y. Jun, and W. Najjar(2004), "A way-halting cache for low-energy high-performance systems," in Proc. Int. Symp. Low Power Electron. Design, pp. 126–131.
3. D. Nicolaescu, A. Veidenbaum, and A. Nicolau(2003), "Reducing power consumption for high-associativity data caches in embedded processors," in Proc. Design, Autom., Test Eur. Conf. Exhibit., pp. 1064–1068.
4. J. Li and Y. Hwang(2005), "Snug set-associative cache: Reducing leakage power while improving performance," in Proc. Int. Symp. Low Power Electron. Design, pp. 345–350.
5. K. Inoue, T. Ishihara, and K. Murakami(1999), "Way-predicting set-associative cache for high performance and low energy consumption," Int. Symp Low Power Electron. Design, pp. 273–275.
6. L. Jin and S. Cho(2006), "Reducing cache traffic and energy with macro data load," in Proc. Int. Symp. Low Power Electron. Design,pp. 147–150.
7. M. D. Powell, A. Agarwal, T. N. Vijaykumar, B. Falsafi, and K. Roy(2001),"Reducing set-associative cache energy via way prediction and selective direct-mapping," in Proc. Int. Symp. Microarchit., pp. 54–65.
8. S. Segars(2001), "Low power design techniques for microprocessors," in Proc.Int. Solid-State Circuits Conf.
9. T. Ishihara and F. Fallah(2005), "A way memoization technique for reducing power consumption of caches in application specific integrated processors," in Proc. Design Autom. Test Eur. Conf,pp. 358–363.
10. D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in Proc. Int. Symp.Comput. Archit., Jun. 2000, pp. 83–94.