# An Energy Saving Tag Cache Model

S.Subha
SITE
VIT, Vellore, India

## ABSTRACT
Energy consumption in caches is widely studied topic. The access to a cache line consumes energy. This paper proposes exclusive cache model that reduces the energy consumption over the tag cache model. The proposed model assumes two level exclusive cache with tag cache in level one. The tag cache consists of tag information of all cache levels. It is stored in cache in level one. An address is checked in tag cache and the corresponding line is accessed. On miss, the line is placed as in exclusive cache case updating the tag cache. The proposed model compares subsets of tag during address mapping. This turns on selectively the comparison circuitry in tag cache selectively saving energy consumption. A mathematical model is developed for the proposed model The proposed model is simulated with SPEC2000 benchmarks. The average memory access time is comparable with tag cache with energy saving of 7%.

## General Terms
Cache memory

## Keywords
Average memory access time, Energy consumption, Set associative cache, tag cache

## 1. INTRODUCTION
Exclusive caches find wide application in multi cores. A cache line is present in only one cache level in exclusive cache. Energy consumption in caches is a topic of interest in present day. A model is proposed in [4] where the level two cache has tag entries of level three cache. The tag cache model is a cache model in which only one cache way is enabled. This reduces the cache energy consumption. The traditional exclusive cache is assumed in this model. In addition the level one cache has another cache called tag cache. This cache has the tag values of all the cache ways in all levels. An address is matched with tag array at various levels. On a match, the corresponding cache way is accessed. On miss, the exclusive cache replacement policy proposed in [3] is implemented updating the tag cache suitably. In this model, only the tag cache along with corresponding cache way is enabled on hits. The tag cache along with ways in the corresponding sets in all cache levels are enabled on conflict misses. This reduces the energy consumption. This paper proposes an algorithm for tag cache matching in this model. The bits in tag cache are matched selectively starting from the least significant bits due to spatial locality. This turns on selectively the circuitry for compare logic. The proposed model is simulated with SPEC2K benchmark. Energy saving of 7% is observed for the simulations with no change in average memory access time (AMAT) when compared with tag cache model.

The rest of the paper is organized as follows. Section 2 gives the motivation, section 3 proposed model, section 4 performance analysis, section 5 simulation, and section 6 acknowledgements, section 7 conclusion.

## 2. MOTIVATION

Consider two level exclusive cache. Level one cache is two way set associative with two sets. Level two cache is four way set associative with two sets. Consider the address trace 0,2,4,6,4,6,0,2. Consider the tag cache algorithm. According to this algorithm there is another cache at level one. Let us call this cache as tag cache. It holds the tag information of all the ways in all cache levels. Assume all caches have same line size.

1. Check if a line exists in any cache level by probing into tag cache. If found, it is cache hit, access the line and stop.
2. If the cache line is not present in any cache level, it is cache miss. Place the line in level one cache according to exclusive cache algorithm proposed in [3]. Update the tag cache entries.
3. Stop.

According to the above algorithm, the lines in the address trace are placed as in exclusive cache. However, to place 0, 2 only level one cache is accessed. To place 4, 6 cache levels one and two are enabled. To access 4, 6 as cache hits, cache level one is enabled. To access 0, 2 as cache hits, cache level two is enabled. The tag array is enabled for all accesses. For placing addresses 4, 6 two cache entries in tag cache is enabled. For the rest only one cache entry is enabled. This reduces the lines enabled in the cache system. Assume that the cache system operates in two modes – high power mode and low power mode. The cache is in low power mode when not accessed. Let the size of address be 32 bits. Let the cache line size be 32bytes. The number of bits for indexing into a set in level one and level two cache is one bit. The number of bits in the tag during cache address mapping is 32-(5+1) =26bits. The number of entries in the tag cache is 2*2+4*2=12. The number of cache lines in the tag cache is (12*26)/ (32*8) which is approximately two. Let the energy consumed be five joules in low power mode and fifteen joules in high power mode per cache line. The energy consumed by the tag cache is 2*15J = 30J. The energy consumed by the level one and level two cache in low power mode for entire address trace is

8*(2*2+4*2)*5J = 480J. The additional energy consumed is calculated as follows.

1. For access of 0, 2 number of enabled ways = 2
2. For access of 4, 6 number of enabled ways = 2+4 = 6
3. For access of 4, 6 number of enabled ways = 1
4. For access of 0,2 number of enabled ways = 1

The additional energy consumed for access of the given address stream is (2*2+2*6+2*1+2*1)*10J = 200J. The total energy consumed is given by 30J+480J+200J=710J

Next consider the following algorithm.

1. Check if a line exists in any cache level by probing into tag cache selectively. This is done by choosing bits from least significant bit in some groups say for example 2 bits, 4 bits etc. On a mismatch in one group, do step 2. On match in all groups , it is cache hit, access the line and stop.
2. If the cache line is not present in any cache level, it is cache miss. Place the line in level one cache according to exclusive cache algorithm proposed in [3]. Update the tag cache entries.
3. Stop.

The simulation for the same address trace with the above algorithm is as follows. There are 26 bit in the tag entry. Assume the tag check is performed from least significant bit (LSB) in 2, 4, 4, 4, 4, 4, 4 bits. The total number of circuit enabling for this combination is 38,11,5,5,5,5,5 for these bits taken cumulatively successively. Assuming energy consumed is five Joules in low power mode and fifteen Joules in high power mode per cache line, the energy consumed in high power mode per bit is 0.059J. The energy consumed in low energy mode in cache system for entire address trace is 8*(2+2*2+4*2)*5J= 448J. The additional energy consumed in the cache system excluding tag cache is 200J. For the tag cache the energy consumed in high power mode is given by (38+11+5+5+5+5+5)*0.0595J=4.403J. The total energy in the cache system is equal to (448+200+4.403)J=652.403J. There is savings in energy consumption of 8%. This is the motivation of this paper.

## 3. PROPOSED MODEL

Consider a tag cache system with two cache levels. Both cache levels are set associative caches. The number of ports is one in both caches. Levels one and two are set associative cache with associativity of $w_1$ and $w_2$ respectively. Let the size of level one cache is $S_1$, level two cache is $S_2$. The line size is same in all cache levels. Let the number of sets in level one and level two cache be $s_1, s_2$ respectively. The tag cache is another level one cache. Let the tag in level one and level two caches be placed in contiguous locations in the tag cache. It is assumed that the line size is equal to the capacity of the larger of sets in the two level cache. Thus one line is accessed to access all the tag entries in a set. The access time for the tag cache is assumed to be equal to level one cache access time. The level one cache access accesses the tag and data. Due to the assumptions made on line size of tag cache, the tag cache access time is reasonable. The first $w_1 s_1$ entries in tag cache represent the tag information of the level one cache. Similarly the rest of the elements of the array hold the tag information of level two cache. Let the tag size be T blocks. This is shown in Figure 1. Consider address a. This has to be mapped to a cache line. Consider the following algorithm

1. Compute the following.

Set1 = a mod $S_1$

Tag1 = a div $S_1$

Set2 = a mod $S_2$

Tag2 = a div $S_2$

2. Check in tag cache for the matching of Tag1. During this process check for subset of bits to match starting with LSB i.e. classify the bits in the tag cache into groups and check for group matching with Tag1. If a group of bits are identical proceed with next group else abort the process as mismatch. If all the groups in tag cache are identical with Tag1, the line is present in level one cache. This is a level one hit. Access level one cache and stop. If there is no match in level one cache check for match of Tag2 using similar procedure. If a match is found, it is level two cache hit. Access the level two cache line and stop.
3. This is a miss condition. Place the least recently used line in Set2 of level two cache in main memory. Transfer the least recently used line of Set1 in level one cache in the evicted level two line. Place the line with address a in level one cache. Update the entries in the tag cache.
4. Stop.

The above algorithm enables a cache line in level one or higher levels only if there is a cache hit or in the case of a miss in all cache levels. The cache is exclusive in nature. The proposed model has scalability.

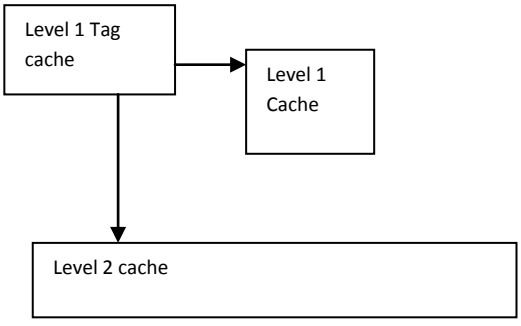


Figure 1 An Exclusive Cache Architecture

## 4. PERFORMANCE ANALYSIS

Consider the system described in Section 3. Denote $C_{prop}$ to represent the proposed system, $C_{tag\_excl}$ the tag cache system. The parameters for the proposed system are given in Table 1. The average memory access time (AMAT) for the proposed system is given by

$$AMAT\left(C_{prop}\right)=\frac{1}{R}\begin{pmatrix} Rt_0+h_1t_1+h_2t_2+ \\ cmiss1\left(t_1+2t_0\right)+ \\ cmiss2\left(t_2+3t_0\right)+ \\ miss\begin{pmatrix} 4t_0+t_1+t_2+ \\ t_{12}+t_{2m}+t_{1m} \end{pmatrix} \end{pmatrix} \quad (1)$$

The first term in (1) is to access the tag cache. The second and third terms are the hit time access to level one and level two caches respectively. The fourth term is the time taken to fill empty way in level one cache. This involves accessing tag cache in level one, fetching the line to level one cache and updating the tag cache entry. The fifth term is the time taken to fill empty way in level two cache. This includes accessing the tag cache to check for match in level one cache and level two cache and placing the line in level two cache, updating the tag cache entry. The sixth term is the time taken to bring in a line in fully filled level one and level two cache. This involves checking for tag match in level one cache, level two cache, replacing the level one cache line and updating the tag cache entries. As the tag cache contains the tags in consecutive locations, it may be the case that the tag entries in level one and level two are in two different cache blocks. The AMAT for the tag cache is also the same.

$$AMAT\left(C_{tag\_excl}\right)=\frac{1}{R}\begin{pmatrix} Rt_0+h_1t_1+h_2t_2+ \\ cmiss1\left(t_1+2t_0\right)+ \\ cmiss2\left(t_2+3t_0\right)+ \\ miss\begin{pmatrix} 4t_0+t_1+t_2+ \\ t_{12}+t_{2m}+t_{1m} \end{pmatrix} \end{pmatrix} \quad (2)$$

The AMAT for both the caches are same.

Table 1 Definitions of parameters in proposed system

| **Parameter** | **Description** |
|---|---|
| R | Total references |
| $h_1$ | Hits in level one cache |
| $h_2$ | Hit in level two cache |
| cmiss1 | Misses filled in Empty level one cache |
| cmiss2 | Misses filled in Empty level two cache |
| miss | Conflict misses in level one and level two cache |
| $t_0$ | Time to access tag cache |
| $t_1$ | Time to access level one cache |
| $t_2$ | Time to access level two cache |
| $t_{1m}$ | Transfer time between main memory and level one cache |
| $t_{2m}$ | Transfer time between main memory and level two cache |
| $t_{12}$ | Transfer time between level one and level two cache |
| $w_1$ | Level one cache associativity |
| $w_2$ | Level two cache associativity |

For a cache hit, the energy consumed is calculated as follows. It is assumed that the cache operates in two modes – high power mode and low power mode. On access to cache way, its corresponding set is placed in high power mode. Else, it is in low power mode. Let $E_{high}$ , $E_{low}$ be the energy consumed by one bit in the cache way in the proposed model in high power mode and low power mode respectively. Let $W_{high}, W_{low}$ be the energy consumed by one bit in cache way in tag cache in high power mode and low power mode respectively. Let $E_{delta,} W_{delta}$ be the difference in energy level in cache way and tag cache way per bit between the two modes of operation. When no cache operation is performed, the energy consumed in the cache system is $(w_1 s_1 + w_2 s_2)8LE_{low} + 8TLW_{low}$ where L is the line size in bytes. It is assumed that the line size is same for all cache levels for this discussion. Let m be the number of groups of bits in the tag for tag comparison. Assume the groups are of same number of bits g. Let $b_1, b_2, .., b_m$ be the number of times each of the m group of bits are compared in address trace of R references. The additional energy consumed for tag comparison is $\left(\sum_{i=1}^{m} b_i\right) gW_{delta}$ The additional energy consumed for the R references is apart from the tag comparison is

$$\begin{aligned} &h_1(w_1 E_{delta})8L + h_2(w_2 E_{delta})8L + \\ &cmiss1(w_1 E_{delta} + W_{delta})8L + \\ &cmiss2(w_2 E_{delta} + 2W_{delta})8L + \\ &miss(w_1 E_{delta} + w_2 E_{delta} + 2W_{delta})8L \end{aligned} \quad (3)$$

Hence the total additional energy consumed is given by

$$\begin{pmatrix} h_1(w_1 E_{delta})8L + h_2(w_2 E_{delta})8L + \\ cmiss1(w_1 E_{delta} + W_{delta})8L + \\ cmiss2(w_2 E_{delta} + 2W_{delta})8L + \\ miss(w_1 E_{delta} + w_2 E_{delta} + 2W_{delta})8L + \\ gW_{delta}\sum_{i=1}^{m} b_i \end{pmatrix} \quad (4)$$

The first term in equation (4) is for the hits in level one cache. Here one set in level one cache is accessed. The second term is for hits in level two cache. The third term is the additional energy consumed for placing in free level one cache. The tag cache entry is updated in this case. The fourth term is for placing a line in free level two cache. The fifth term is the energy consumed for conflict misses. The last term is the energy consumed for tag comparison. Consider a program with R references. The total energy consumed for this address trace is given by

$$\begin{pmatrix} R(8LE_{low}(w_1 s_1 + w_2 s_2) + 8LTW_{low}) + \\ h_1(w_1 E_{delta})8L + h_2(w_2 E_{delta})8L + \\ cmiss1(w_1 E_{delta} + W_{delta})8L + \\ cmiss2(w_2 E_{delta} + 2W_{delta})8L + \\ miss(w_1 E_{delta} + w_2 E_{delta} + 2W_{delta})8L + \\ gW_{delta}\sum_{i=1}^{m} b_i \end{pmatrix} \quad (5)$$

The first term is the energy consumed when the cache is not accessed. The rest of the terms are the additional energy consumed during cache operation.

Consider the tag cache model. The energy consumed in this model is given by

$$\begin{aligned} &R(S_1 + S_2)8LE_{low} + 8RTLW_{low} + \\ &h_1(W_{delta} + w_1 E_{delta})8L + \\ &h_2(2W_{delta} + w_2 E_{delta})8L + \\ &cmiss1(W_{delta} + w_1 E_{delta})8L + \\ &cmiss2(2W_{delta} + w_2 E_{delta})8L + \\ &miss(2W_{delta} + w_1 E_{delta} + w_2 E_{delta})8L \end{aligned} \quad (6)$$

A saving in energy consumption is observed when

$$\begin{pmatrix} R(8LE_{low}(w_1 s_1 + w_2 s_2) + 8LTW_{low}) + \\ h_1(w_1 E_{delta})8L + h_2(w_2 E_{delta})8L + \\ cmiss1(w_1 E_{delta} + W_{delta})8L + \\ cmiss2(w_2 E_{delta} + 2W_{delta})8L + \\ miss(w_1 E_{delta} + w_2 E_{delta} + 2W_{delta})8L + \\ gW_{delta}\sum_{i=1}^{m} b_i \end{pmatrix}$$

<=

$$R(S_1 + S_2)8LE_{low} + 8RTLW_{low} + h_1(W_{delta} + w_1E_{delta})8L + h_2(2W_{delta} + w_2E_{delta})8L + cmiss1(W_{delta} + w_1E_{delta})8L + cmiss2(2W_{delta} + w_2E_{delta})8L + miss(2W_{delta} + w_1E_{delta} + w_2E_{delta})8L \quad (7)$$

The energy consumed in the cache alone without the tag cache in high energy mode is given by

$$R((w_1s_1 + w_2s_2)8LE_{low} + 8TLW_{low}) + h_1(w_1E_{delta})8L + h_2(w_2E_{delta})8L + cmiss1(w_1E_{delta})8L + cmiss2(w_2E_{delta})8L + miss(w_1E_{delta} + w_2E_{delta})8L + gW_{delta}\sum_{i=1}^{m} b_i \quad (8)$$

It can be inferred that as the size of the number of bits per slice in tag comparison decreases energy efficiency increases. The optimal case would be to match the tag bits one bit at a time which would increase the AMAT.

## 5. SIMULATION

The proposed model was simulated on SPEC2K benchmark on uniprocessor system. Simplescalar binaries were built for the SPEC2K benchmarks. Addresses were collected from these binaries. The address trace was simulated on the proposed model. The parameters for the simulation are given in Table 2.

The proposed model was compared with the tag cache model. It is assumed that the energy consumed in low power mode is 5J and high power mode is 15J for one cache way in this simulation. The size of one cache way is 32 bytes for the cache line and 26 bits for the tag entry. The energy consumed in low energy mode is 5J/282 per bit, in high energy mode is 15J/282 per bit the difference being 10J/282 per bit. The author calculated the average memory access time and energy consumed. For the AMAT, the hits in two cache levels, the placement of line in empty, full cache line were gathered by C routines. The average memory access time is almost same in both the models. The AMAT values are given in Table 3 and Figure 2. The energy consumed is given in Table 4 and Figure 3. The energy is saved by 7% as seen from Figure 3 when compared with tag cache model. The energy consumption depends on the number of bits in each group during comparison of the tag. The optimal procedure would be to compare one bit at a time at the cost of increased AMAT.

Table 2. Simulation Parameters

| S.No | Parameter | Value |
|---|---|---|
| 1 | Level one cache size | 32KB |
| 2 | Level one associativity | 4 |
| 3 | Level two cache size | 32KB |
| 4 | Level two associativity | 64 |
| 5 | Level one access time | 3 cycles |
| 6 | Level two access time | 18 cycles |
| 7 | Level one to level two transfer time | 18 cycles |
| 8 | Level one to memory access time | 60 cycles |
| 9 | Level two to memory access time | 90 cycles |
| 10 | Line size | 32 bytes |

Table 3 AMAT comparison

| name | AMAT(prop) | AMAT(trad) |
|---|---|---|
| 256.bzip2 | 127.6092 | 127.6185 |
| 181.mcf | 21.46317 | 21.46317 |
| 197.parser | 63.70157 | 63.70157 |
| 300.twolf | 21.01462 | 21.01462 |
| 255.vortex | 15.75037 | 15.75037 |
| 175.vpr | 20.43262 | 20.43262 |

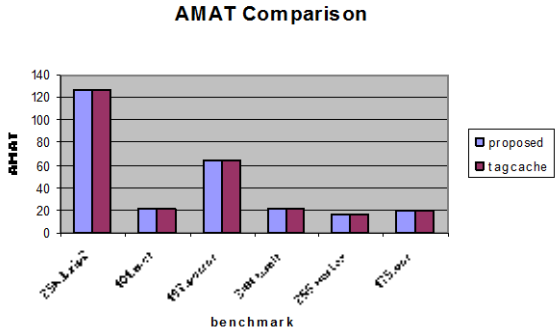


Figure 2 AMAT comparison

Table 4 Energy Comparison

| name | E(prop) | E(trad) | %improve |
|---|---|---|---|
| 256.bzip2 | 4739.053 | 5320.195 | 10.92331 |
| 181.mcf | 4673.922 | 5054.567 | 7.53071 |
| 197.parser | 4695.78 | 5249.914 | 10.55511 |
| 300.twolf | 4705.502 | 5057.634 | 6.962397 |
| 255.vortex | 4755.494 | 4946.852 | 3.868285 |
| 175.vpr | 4853.68 | 5106.585 | 4.952526 |

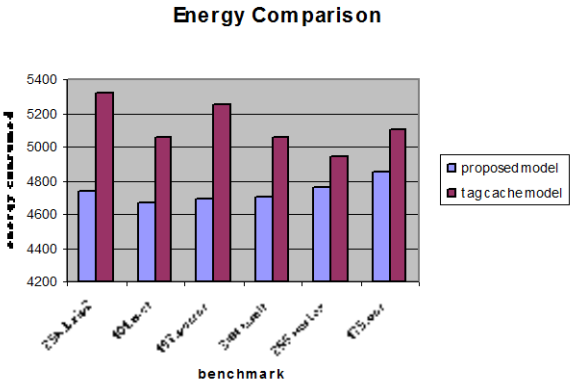


Figure 3 Energy Comparison

## 6. ACKNOWLEDGEMENTS

The author expresses thanks to Santa Clara University, CA, USA for providing SPEC2000 benchmarks.

## 7. CONCLUSION

An exclusive cache model that saves energy consumption is proposed in this paper. The model assumes tag cache at level one to contain the tags of the lines in all cache levels. An address is matched with the tags in this cache. The tag bits are compared in slices. This is as proposed in literature to reduce the energy consumption. On a hit, the corresponding cache line is accessed. On a miss the line is placed in level one cache replacing level one and level two cache lines respecting exclusive nature. Simulations show that in the proposed model the AMAT is comparable with tag cache with 7% improvement in energy savings.

## 8. REFERENCES

[1] D.A.Patterson and J. L. Hennessy, COMPUTER ARCHITECTURE: A QUANTITATIVE APPROACH, 3rd edition, Morgan Kaufmann Publishers, 2003

[2] N.P.Jouppi, "Improving Direct-Mapped Cache Performance by the addition of a small fully-associative cache and prefetch buffers", *Proceedings of the 17th Annual Symposium on Computer Architecture*, Seattle, WA, USA, *pp 364-373*, August, 1990

[3] N.P.Jouppi and S.J.E.Wilton, "Tradeoffs in Two-level on chip caching" Proceedings of the 21ST annual international symposium on Computer architecture, Chicago, IL, USA, pp. 34-45, April, 1994

[4] L.Zhao, R.Iyer, S.Makineni, D.Newell, L. Cheng, " NCID: a non-inclusive cache, inclusive directory architecture for flexible and efficient cache hierarchies", Proceedings of the 7th ACM international conference on Computing frontiers, Bertinoro, Italy , pp. 121-130, May 17-19, 2010

[5] S.McFarling, "Cache Replacement with DynamicExclusion", Proceedings of ISCA, GoldCoast, Queensland, Australia, pp. 191-200, May 1992

[6] Y.Zheng, B.T.Davis and M.Jordan: "Performance Evaluation of Exclusion Cache Hierarchy", IEEE International Symposium on Performance Analysis of Systems And Software, Austin, Texas, USA, pp. 89-96, March 10-12, 2004.