A Note on Linear based Set Associative Cache address System

Chandramowliswaran N[1], Srinivasan.S[1] and Chandra Segar.T[2]

[1]School of Advanced Sciences
VIT University
Vellore, India
ncmowli@hotmail.com and smrail@gmail.com

[2]School of Information Technology and Engineering
VIT University
Vellore, India
chandru01@gmail.com

Abstract

Based on the system request or application request a set of word is loaded on the *cache memory*. When the system is switched off the cache history gets abscond. The performance of CPU is based on the factors such as *cache hit, write through cache, write back, cache memory mapping technique, CPU speed, bandwidth, cache memory size* etc., Some of the standard cache addresses mapping techniques are *set associative, associative and direct mapping technique*. This paper proposes a novel idea of *set associative cache address mapping* using linear equation. The standard *set associative mapping* is remapped with linear set associative for to secure the data in a non sequential portion by having the standard mapping execution time. This is mainly focus on to design the cache enhancement and improvement.

Keywords: *Cache mapping, physical addressing, associative mapping*

1. Introduction

To measure *set associative cache* address mapping system using linear equation and to reduce time completion of cache address mapping system. The proposed linear cache mapping system aims to understand easily on *associative cache* address mapping and to access fast in cache address mapping system.

In the computer architecture, some of the standard cache addresses mapping techniques are *direct mapping, set-associative mapping, and fully associative mapping.* Generally, the cache mapping techniques differs with other through the way data is fetched and stored from the main memory onto cache and correspondingly. The data is referred from the cache through the policy called Locality of reference such as temporal and spatial reference. The temporal reference of a data refers a period of availability of data with respect to time and the spatial reference of a data refers to the location over the cache. The main aim of this paper is to experiment the time convolution of novel *associative mapping* with respect to standard associative mapping. Initially the central processor makes an effort to stare for a data or a word on the cache point called hit, or else then stare with an extra time as penalty on main memory level called miss. To compute a process on the system, the processor loads the set of passive instructions from the auxiliary memory to system memory to make it active for execution. In this regard, processor has to examine the sufficient space on both cache and physical memory. After the load process, the partial or complete execution of a process is stored on this memory and these changes reflect either only on cache or both cache and physical memory i.e., write policy selection. The *associative memory* refers the physical memory based on the three attributes such as *tag, block and word.* The proposed linear cache mapping remaps the actual reference with the standard one in the linear order fashion.

For instance, the system has the physical memory size of 8 MByte (2^{23}), the cache size 1024 KByte (2^{20}), and the block size is of 128Byte (2^7). From the width of the SET field the number of rows is to be determined. Each set maximum refers to four blocks i.e., *4-way set-associative cache* is applied onto the system. Thus the system consists of:

Number of Blocks in Physical memory
= Physical memory capacity / Per Block Size
= 2^{23} / 2^7 = 2^{16} blocks
Number of Sets = Number of Blocks / Number of ways

$= 2^{16} / 2^2 = 2^{14}$ sets

TAG Size = Physical size / Cache Size

$= 2^{23} / 2^{20} = 2^3$

OFFSET Size = Physical size / (Number of Sets * TAG Size)

$= 2^{23} / (2^{14} * 2^3) = 2^6$. Hence, the Linear System fields are *OFFSET* (= 6) and *TAG* (= 3).

2. Motivation and Background

N. Chandramowliswaran et al., [1] describes that, the exponential growth of Information System needs a vital requirement to protect those data substantially from the prevention of unethical activities. To avoid such scenario in the internal memory system they are proposing a novel technique for *associative mapping* using *Graceful Code (GC) technique.* The processor performance is directly depends on the *cache, mapping technique, bandwidth, front side bus.* In paper [1], their proposed work is focused on secured mapping over *GC* technique applied to acquire a demanding result.

John S. Harper et al., [2] describes Cache behavior is complex and also unstable, but it is a critical factor affecting program performance. Quantitative predictions of miss-ratio and information to guide optimization of cache are required to evaluate cache. Cache simulation gives accurate predictions of miss-ratio, but little to direct optimization. Hence, the program execution time always lesser than simulation time. Many analytical models have been made, but concentrate mainly on direct-mapped caches, often for specific types of algorithm, to give qualitative predictions. Analytical models of cache are presented, applicable to numerical codes consisting mostly of array operations in looping constructs. *Set associative caches* are determined, through an extensive hierarchy of cache reuse and interference effects, including numerous forms of temporal and spatial locality. An advantage is that it indicates sources of cache interference. The accuracy validated through program fragments. The predicted miss-ratios are compared with simulations it will be within 15 percent. The evaluation time of the models is independent it depends upon the problem size, in general many orders of magnitude faster than simulation.

Stefano Di Carlo et al., [3] describe embedded microprocessor cache memories affect from observability and controllability creating problems during in system tests. So they apply procedure to transform traditional march tests into software-based self-test programs *for set associative cache memories* with LRU replacement. In microprocessor testing instruction caches represents a major diffculties due to limitations ing n two areas: 1) test patterns which must be composed of valid instruction opcodes and 2) test result observability : the results can only be observed through the results of executed instructions. For these purpose the proposed system will concentrate on the implementation of test programs for instruction caches. The main part of this work lies in the possibility of applying state-of-the-art memory test algorithms to embedded cache memories without any hardware or performance overheads and guarantee that detection of typical faults arising in nanometer CMOS technologies. The results got by constructing test programs for the LEON3 microprocessor show that it is possible to Protect the fault coverage of the original march tests. The results also consider control blocks of the cache such as validity bits and control circuits, providing reasonable coverage also on these blocks. Additional fields that might be included in a microprocessor cache have not been work. Similarly to validity bits, their coverage should be tested the specific implementation of the target cache memory and whenever required, the test program should be improved to cover undetected faults.

S.Subha et al., [4] describe method to save energy in *set associative* additional information about next access to maintained in the cache ways. All the ways of the cache are put in either disable mode or low energy mode as supported by the cache. *Set associative* has a additional column called next_access_counter which contains the time of the next access of the address. The time counter is maintained to keep the track of the time cycle. During this mapping all the ways of the mapped set are enabled as in a traditional *set associative cache*. The model was simulated on SPEC2000 benchmarks with average energy savings of 19% and performance degradation of about 10%.A cache model for set associative cache that minimizes the energy consumed is proposed in this paper.

Ramy E. Aly et al., [5] describes that Variable-way set associative cache is used to maximize the cache performance and reduce the power consumption with the same performance. The *set-associative cache* Variable-way *set-associative* can be used in High performance or low-power operation modes. The proposed architecture is simulated on simplescalar simulator and tested on several Spec2000 Benchmarks. The results show on average 2% reduction in the miss rate at the high-performance mode and up to 43% reduction of the power consumption at low-power mode. We found that some sets get no performance improvement by doubling the associativity. In variable way *set associative* on sets that only get hit rate improvement.

S. Subha et al., [6] describes *set associative caches* have fixed number of sets with fixed number of ways in each set. The mapping of an address to a set is expanded to a subset of the number of sets by XOR' ing the address with 0, 1, 2, 4..., (Number of sets/2). The line is placed in any of the available ways in this mapping. If the cache is full, the cache is like *set associative cache*. In this mapping the number of ways that a line can be placed in a partially filled cache is increased. This paper proposes a modified address translation procedure in *set associative cache. Set associative* an average memory access time is 13%. The degradation of average memory access time for set associative memory is 256.bzip2. The consumption of energy increases as a function of the cache size.

3. Linear Set associative Mapping

The proposed is deal with the implementation of linear based set associative cache address mapping system using linear equation $y = (\alpha x + \beta) \bmod \boldsymbol{n}$ with the help of *C language.*

SOLUTION METHODOLOGY

In this proposed system, the concept of *set associative mapping* techniques associated with *direct and fully associative mapping*. The cache lines are grouped into sets. The number of lines, ***'n'*** in set can vary from 2 to 16. Set associative mapping will be divided into three parts.

TAG	***SET***	***OFFSET***

ALGORITHM

STEP 1: Initialize all variables to *x, α, β.*
STEP 2: Get the inputs for α, *β* from the user.
STEP 3: If the *G.C.D.* $(\alpha, \boldsymbol{n}) = 1$,
Then perform, $y = \alpha x + \beta \bmod \boldsymbol{n}$ and
set $y_1 = y \bmod n$.

STEP 4: Else print the input values are not satisfied.
STEP 5: End.

The set of single digit valid strings, *'V'* for the given input *x [0],…, x[15]* are as follows:

V[α, β] = { ((1,0), (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (1,7), (1,8), (1,9), (3,0), (3,1), (3,2), (3,3), (3,4), (3,5), (3,6), (3,7), (3,8), (3,9), (5,0), (5,1), (5,2), (5,3), (5,4), (5,5), (5,6), (5,7), (5,8), (5,9), (7,0), (7,1), (7,2), (7,3), (7,4), (7,5), (7,6), (7,7), (7,8), (7,9), (9,0), (9,1), (9,2), (9,3), (9,4), (9,5), (9,6), (9,7), (9,8) , (9,9) }

For instance *V[α,β] = V[5,7]*is shown in Table 1.

Table 1:

X	***Y***	Y_1
0	7	7
1	12	12
2	17	1
3	22	6
4	27	11
5	32	0
6	37	5
7	42	10
8	47	15
9	52	4
10	67	9
11	72	14
12	77	3
13	82	8
14	87	13
15	92	2

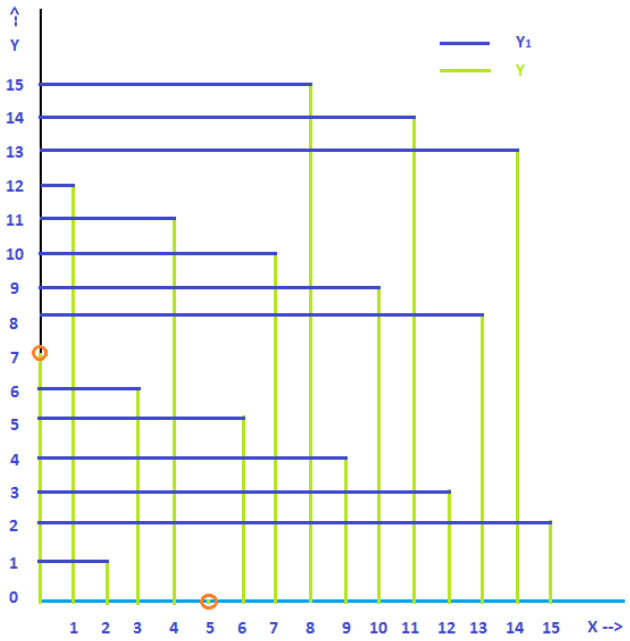
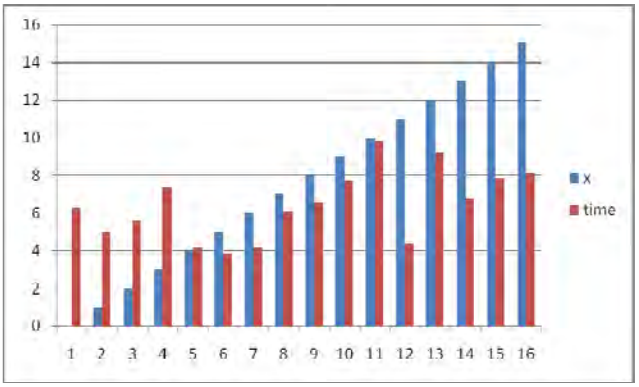


Fig. 1 Remapping Graph of *V [5, 7]*

4. Result



Hence, the above graph clearly states that the searching time of inputs, *X[0],…,X[15]* lies within the linear time of O (n).

5. Conclusions

In this work, linear based set associative cache address mapping has been introduced and implemented by tuning the parameters α, β. Moreover, this work can also be extended by using linear one to one recursive mapping. The time complexity of the existing algorithm takes only O (n).

References

[1] Chandramowliswaran.N, Srinivasan.S, Chandra Segar.T, "A Novel scheme for Secured Associative Mapping," to be submitted.
[2] John S. Harper, Darren J. Kerbyson, and Graham R. Nudd, "Analytical Modeling of Set-Associative Cache Behavior", Compute (1999), Pg. no: 1009-1024.
[3] Stefano Di Carlo, Paolo Prinetto, and Alessandro Savino," Software-Based Self-Test of Set-Associative Cache Memories", Computer (2011), Pg. no: 1030-1044.
[4] S.subha," Set Associative Cache Model with Energy Saving", Information Technology: New Generations (ITNG - 2008), Pg. no: 1249-1250.
[5] Ramy E. Aly, Bharat R. Nallamilli, Magdy A. Bayoumi, "Variable-way Set Associative Cache Design for embedded System Applications," Circuit and system (2003), vol.3., Pg. no: 1435-1438.
[6] S. Subha, "A Set Associative Cache Architecture," Information Technology: New Generations (ITNG – 2010), Pg. no: 1316 – 1317.