International Journal of Research Studies in Electrical and Electronics Engineering(IJRSEEE)
Volume 4, Issue 3, 2018, PP 12-17
ISSN 2454-9436 (Online)
DOI: http://dx.doi.org/10.20431/2454-9436.0403002
www.arcjournals.org

Design Strategy of Cache Memory for Computer Performance Improvement

Tarig Ibrahim Osman Ahmed[1], Elsanosy M. Elamin[2*]

[1]*Assistant Professor, Alrayyan College, Saudi Arabia*

[2]*Department of Electrical Engineering, Faculty of Engineering, University of Kordofan, Sudan.*

***Corresponding Author:** ***Elsanosy M. Elamin**, Department of Electrical Engineering, Faculty of Engineering, University of Kordofan, Sudan.*

Abstract: *In recent technology especially in the industrial fields, the computers are exploited as controlling and monitoring tool to help in system development. As a result, most of the computer problems in various applications are slow speed and poor performance. In this paper, the design strategy was posed to show how to assist in improving both speed and performance. There are many factors that affect the performance of the computer such as the processor speed, the size of RAM, and the weakness of the cache memory strategy for the processor. These factors are the most influential factors on the speed and performance of the processor, which are result in performance deterioration. Most cache memories are designed outside the processor units which are affecting the data transfer speed to/from the processor, delayed processor data access time, and processor access time. The C++ program was employed as simulation tool for performance evaluation to clearly show the effect of the cache memory. The simulation results explicit the great impact of the added cache memory on both the processor speed and the computer performance when the cache was designed inside the processor unit. Also, shows negative results when designing the cache outside the processor unit.*

1. INTRODUCTION

A cache memory, sometimes called a cache store or RAM cache, is fundamentally a portion of memory made of high-speed static RAM (SRAM) instead of the slower and cheaper dynamic RAM (DRAM) used for main memory. Memory caching is effective because most programs access the same data or instructions over and over. The principal position of the cache memory in the computer structure was located in between the main memory and CPU core as shown in figure 1 as simple and minimum cache memory configuration. It corresponds to the architecture which could be found in early systems which deployed CPU caches.

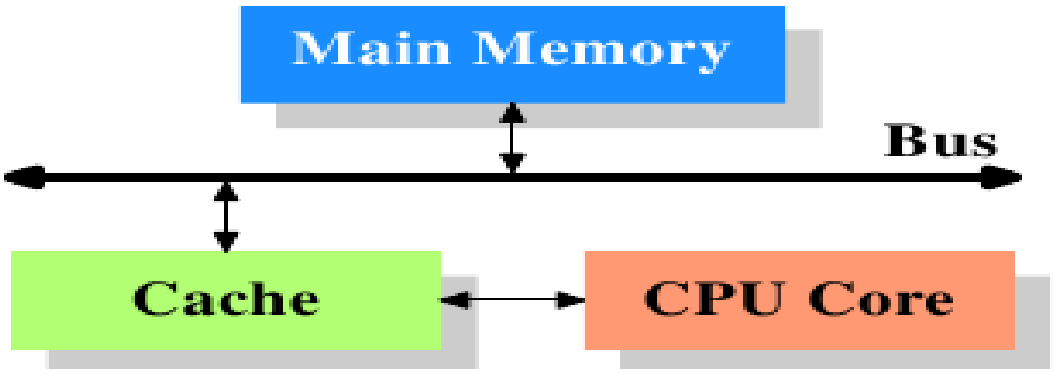

Figure1. *Minimum cache configuration*

The cache memory also developed to multi different levels to enhance the overall processor performance. These levels achieved some development to the overall performance. The simple level that was developed is depicted in figure 2 to illustrate the three levels of cache memory and introduces the nomenclature that will be used in this paper. L1d is define the level 1 data cache, whereas, L1i express the level 1 instruction cache, etc.

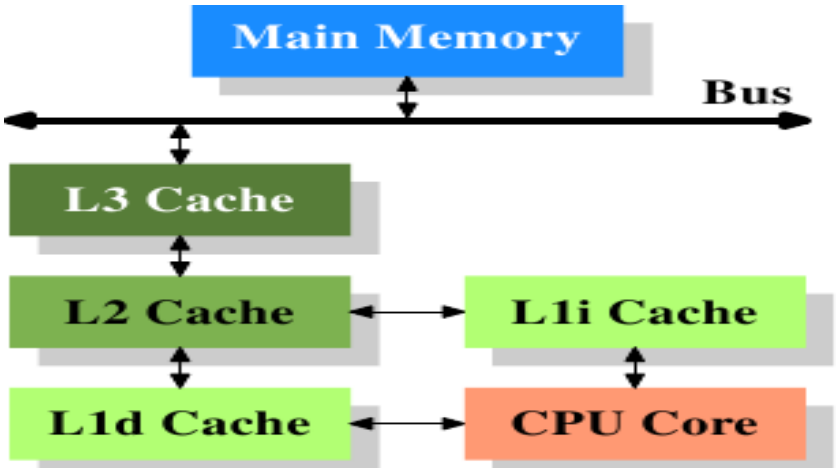

Figure2. *Processor with Level 3 Cache*

1.1. Cache Operation at High Level

When memory content is needed by the processor, the entire cache line is loaded into the L1d. The memory address for each cache line is computed by masking the address value according to the cache line size. For a 64 byte cache line this means the low 6 bits are zeroed. The discarded bits are used as the offset into the cache line. The remaining bits are in some cases used to locate the line in the cache and as the tag. In practice, an address value is split into three parts. For a 32-bit address it might look in figure 3 as follows:

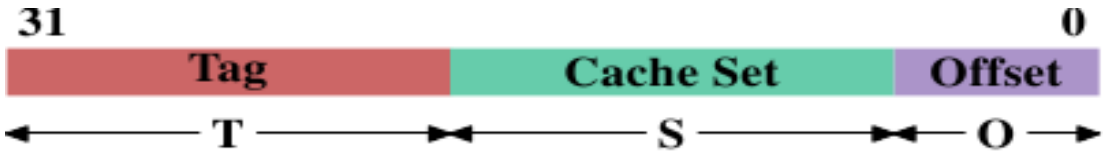

Figure3. *The address value in 32-bit address*

With a cache line size of 2^O the low O bits are used as the offset into the cache line. The next S bits select the "cache set". For now, it is sufficient to understand there are 2^S sets of cache lines. This leaves the top 32 - S - O = T bits which form the tag. These T bits are the value associated with each cache line to distinguish all the *aliases* {*All cache lines with the same* S *part of the address are known by the same alias.*} which are cached in the same cache set. The S bits used to address the cache set do not have to be stored since they are the same for all cache lines in the same set. For more clarification, figure 4 expressed the access time for random data transfer.

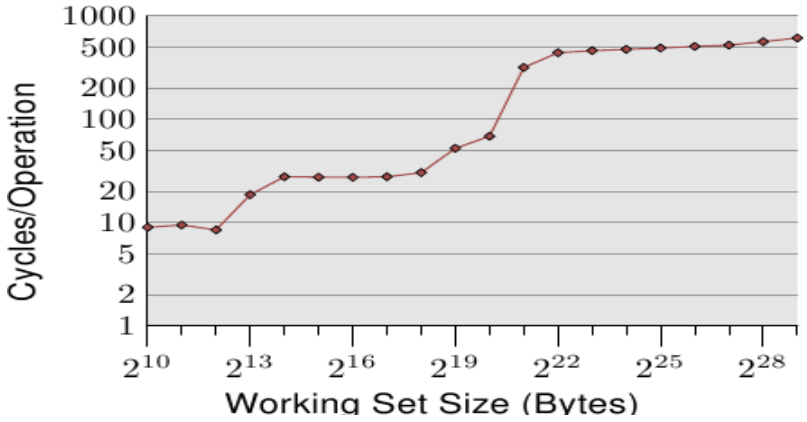

Figure4. *Access Times for Random Writes*

As a result, the cache which is not easily defeated by unfortunate or deliberate selection of addresses with the same set numbers and at the same time the size of the cache is not limited by the number of comparators which can be implemented in parallel. If the cache level grows, as in figure 4, only the number of columns are increases, not the number of rows. The number of rows only increases if the associativity of the cache is increased. Today processors are using associativity levels of up to 16 for L2 caches or higher. L1 caches usually get by with 8.

The addresses are mapped into the cache by using

O = $\log_2$ cache line size

S = $\log_2$ number of sets

Figure 5 makes the data more comprehensible. It shows the data for a fixed cache line size of 32 bytes. Looking at the numbers for a given cache size, it can be noted that associativity can indeed help to reduce the number of cache misses significantly. For an 8MB cache going from direct mapping to 2-way set associative cache saves almost 44% of the cache misses. The processor can keep more of the working set in the cache with a set associative cache compared with a direct mapped cache.

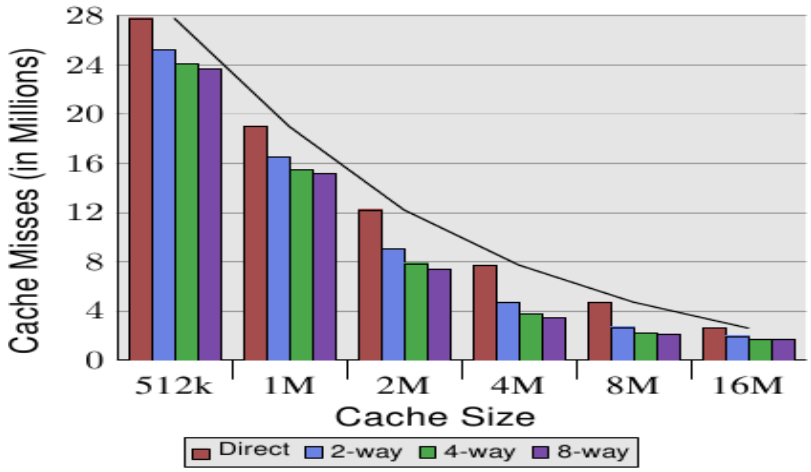

Figure5. *Cache Size versus Associativity (CL=32)*

2. METHODOLOGY

The methodologies that achieve the objectives of this paper based on many steps:

1. Expressing the state transition of multi different process into software program in order to simulate the processing by using C language.
2. Using keypad as input hardware.
3. Using parallel port interface in order to connect the out port with the personal computer.
4. Using IC SN74245 as protection between the keypad and the parallel port interface.

2.1. D-25 Connector

This component is used to interface the computer to the electronic circuit. The female D-25 connector is principally exploited for device control and communication through software program. It consists of data, control, and status lines to be used as input/output buses. these lines are connected to relevant registers as shown in figure 6 [2].

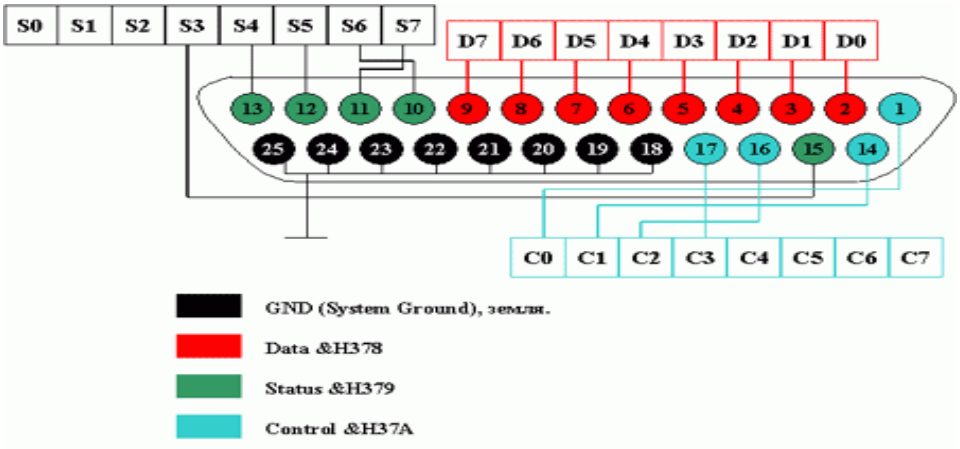

Figure6. *The female d-25 connecter*

2.2. Lab Link Cable

The lab link cable is connected to the computer for processing and display the output.

2.3. Ic Sn74245

The 74245 is a high-speed Si-gate CMOS device. It is an octal transceiver featuring non-inverting 3-state bus compatible outputs in both send and receives directions. It has 20 pin as shown in figure 7.

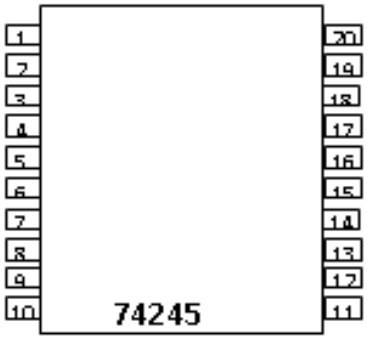

Figure7. *IC SN 74245*

It has an output enable (EN), input for easy cascading and a send/receive (DIR) input for direction control. EN controls the outputs so that the buses are effectively isolated.

2.4. Flow Chart

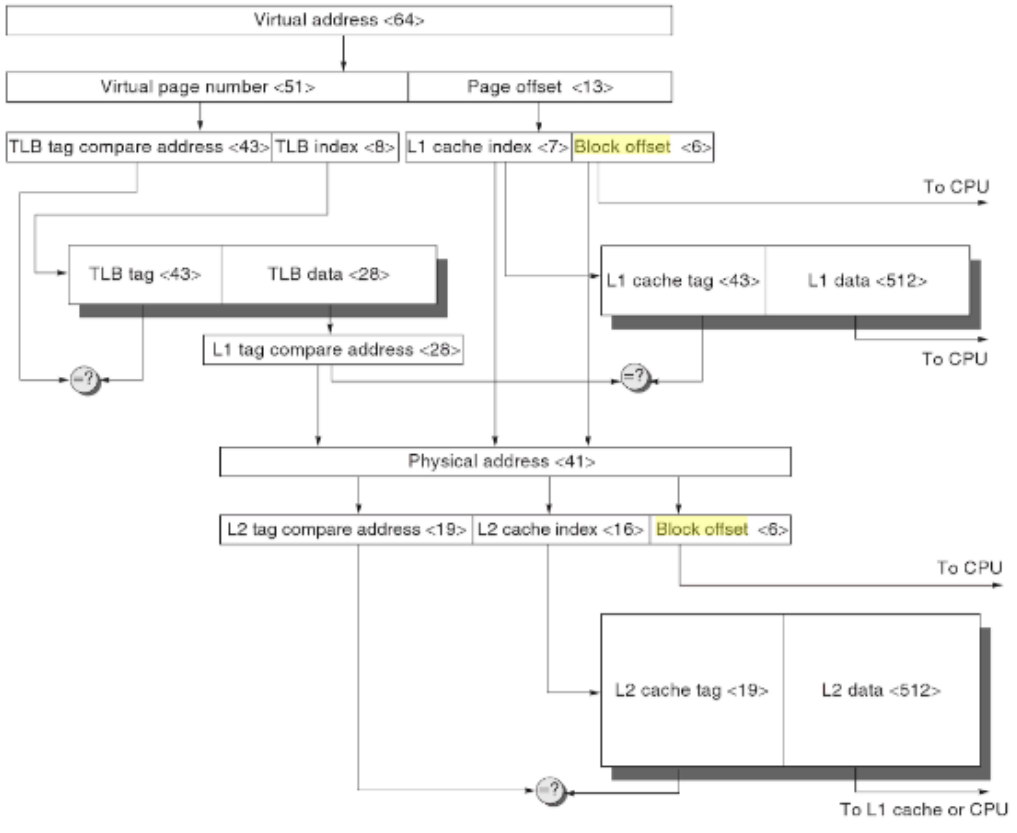

Figure8. *Flow chart*

3. THE SIMULATION RESULT

The simulation processes were arranged and carried out according to the normal parameters and RAM size in order to evaluate the effect of multi different cache levels on both processor speed and overall computer performance. Due to limitations of the available RAM the working set size had to be restricted to 2^{24} bytes which requires 1GB to place the objects on separate pages. The valued simulation results are shown in figure 9.

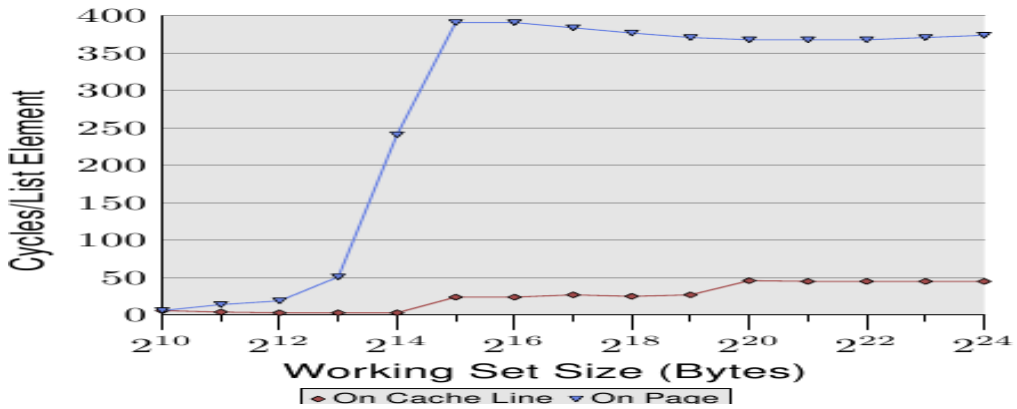

Figure9. *TLB Influence for Sequential Read*

For in depth evaluation, a few more details of the prefetch implementation by looking at the data of test runs where the list elements are modified. Figure 10 shows three lines.

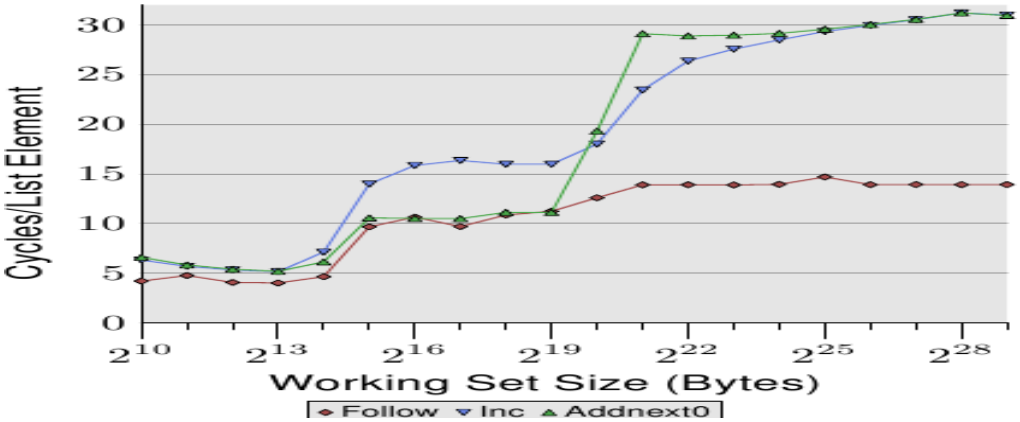

Figure10. *Sequential Read and Write, NPAD=1*

One last aspect of the sequential, efficient cache handling is the size of the cache. This should be obvious but it still should be pointed out. Figure 11 shows the timing for the Increment benchmark with 128-byte elements (NPAD=15 on 64-bit machines). Below are the measurements shows the improvement on processor performance from three different machines.

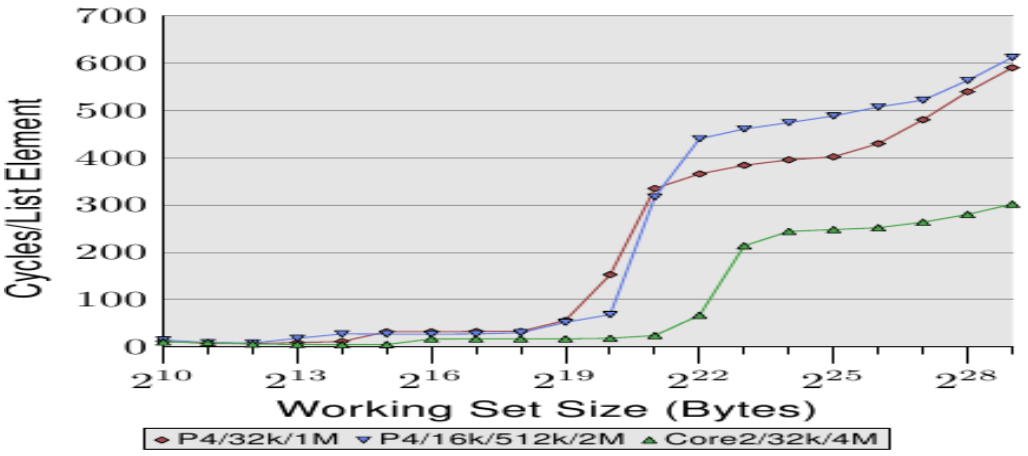

Figure11. *Advantage of Larger L2/L3 Caches*

4. CONCLUSION

The interesting part of the graph is not necessarily how well the Core2 processor performs relative to the other two (although it is impressive). The main point of interest here is the region where the working set size is too large for the respective last level cache and the main memory gets heavily involved. And has shown that of the processor speed were obtained, the computer performance was improved when the cache was designed inside the processor unit, and the negative results when designing the cache outside the processor unit.

REFERENCES

[1] "IBM z13 and IBM z13s Technical Introduction" . IBM. March 2016. p. 20.

[2] Elsanosy M. Elamin, Murtada M. Abdelwahab, Abdelrasoul J. Alzubaidi, Wireless Secured Remote Control System Expansion, IOSR Journal of Engineering IOSRJEN, 5(1), Jan. 2015, 48-51.

[3] John L. Hennessy; David A. Patterson (2011). Computer Architecture: A Quantitative Approach. p. B-9. ISBN 978-0-12-383872-8.

[4] Sanjeev Jahagirdar; Varghese George; Inder Sodhi; Ryan Wells (2012). "Power Management of the Third Generation Intel Core Micro Architecture formerly codenamed Ivy Bridge" (PDF). hotchips.org. p. 18. Retrieved 2015-12-16.

[5] Kaxiras, Stefanos; Ros, Alberto (2013). "A New Perspective for Efficient Virtual-Cache Coherence". 40th International Symposium on Computer Architecture (ISCA): 535–547. doi:10.1145/2485922.2485968.

[6] Agner Fog (2014-02-19). "The micro architecture of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers" . agner.org. Retrieved 2014-03-21.

[7] Ian Cutress (2016-08-18). "AMD Zen Micro architecture: Dual Schedulers, Micro-Op Cache and Memory Hierarchy Revealed". Anand Tech. Retrieved 2017-04-03.

[8] "Intel Iris Pro 5200 Graphics Review: Core i7-4950HQ Tested". AnandTech. Retrieved 2014-02-25.

[9] Tian Tian; Chiu-Pi Shih (2012-03-08). "Software Techniques for Shared-Cache Multi-Core Systems". Intel. Retrieved 2015-11-24.

[10] Ying Zheng; Brian T. Davis; Matthew Jordan (2004-06-25). "Performance Evaluation of Exclusive Cache Hierarchies". Michigan Technological University. Retrieved 2014-06-09.

[11] Aamer Jaleel; Eric Borch; Malini Bhandaru; Simon C. Steely Jr.; Joel Emer (2010-09-27). "Achieving Non-Inclusive Cache Performance with Inclusive Caches" . jaleels.org. Retrieved 2014-06-09.

[12] Mahapatra, Nihar R.; Venkatrao, Balakrishna. "The processor-memory bottleneck: problems and solutions" . Association of Computing Machinery. doi:10.1145/357783.331677. Retrieved 2013-03-05.

[13] "A Survey of Architectural Techniques For Improving Cache Power Efficiency", S. Mittal, SUSCOM, 4(1), 33-43, 2014

AUTHORS' BIOGRAPHY

Tarig Ibrahim Osman Ahmed, Holds a BSc of Electrical and Computer Engineering from Omdurman University 1999. Worked as electronic engineer in Sudanese Currency Factory, and IT engineer in the National Computer Company (NARS), and worked as a teacher at Sudanese-Korean Institute of Technology. Holds a MSc of Electronic and Computer Engineering at Omdurman University 2008. Worked as lecturer at Al-Rayyan High Institute in Saudi Arabia, then as a lecturer at Tiba university, and Arab Open University. Also worked as electronic and IT engineer in the MAKAREM Al Madinah Hotel. Holds PhD of Electronic Engineering from Sudan Academy of Science, Currently an Assistant professor at Al Rayyan colleges, Saudi Arabia

Elsanosy M. Elamin, Obtained his BSc in Electrical and Computer Engineering at Omdurman Islamin University 1999. Received his MSc. in Communication at Karary Academy of Technology 2003 and Ph.D. in Electronics at Sudan Academy of Sciences 2014. Assistant Professor and head department of electrical engineering, University of Kordofan. His field interest is in Electronic Design, Data Communication Networks, and Mobile Communication Networks.

Citation: *Tarig Ibrahim Osman Ahmed, Elsanosy M. Elamin. "Design Strategy of Cache Memory for Computer Performance Improvement", International Journal of Research Studies in Electrical and Electronics Engineering, 4(3), pp 12-17. DOI: http://dx.doi. org/10.20431/2454-9436.0403002*