The Root Tile Design for Level 1 Cache for Non Uniform Architecture

**Suma Sannamani[1], Dr.Manjudevi[2]*
Visvesvaraya Technological University, Belagavi, India
Oxford College of Engineering, Bangalore, India,
sumabs2014@gmail.com,manju3devi@gmail.com

Abstract: NUCA has become solution for wire delay problems, where wire delay problems increases on chip latency in multiprocessor system. Non uniform architecture is used for cache memory. Here cache is divided into tiles ,each tiled cache is accessed with different latency. Hence it is called non uniform. Access data defines search algorithm across architecture. This paper involves design of root tiles which accepts request from processor and forward request to child cache tiles. Here we have used Xilinx simulation tool to analyze the performance.

Keywords: Root tile, NUCA, Xilinx

1.Introduction:

High performance processor uses different level of caches on the processor die. The size of these different level of caches are expected to increase. While choosing size of each level cache designer has to balance access time, area and budget. Designer must consider wire delay which affects access latency. Here work involves tile cache design using direct mapping and the design of root tile which accepts request from processor and forward to level 1 cache.

1.1 Tile Organization

Tile provides interconnection on-chip. Implementation of tiled router involves basic operation forwarding miss request, address needs to be searched. These request and address are sent to cache tile. Search network will forward miss request to next level tile. When tiled cache receives a search request and assumes if cache tile experience miss-data is not found, then it forwards the request. In another case if the cache tile found data to be searched, which is to be considered as hit, it transport data to processor.

Cache Memory is a fast memory. Cache memory used to speed up main memory and also it provide synchronization between main memory with high-speed CPU. Cache memory is cost effective compared to CPU register but compared to RAM it is cost . But there is need to consider advantage of Cache memory speed co pared to fast memory . Cache is placed between RAM and the CPU. cache memory usually holds repeatedly used data and instructions. Hence those data and instructions easily accessible to the core whenever required. Cache memory reduces the time required to read data from the Main memory. The cache memory is a small memory. But speed is more. It stores portion of the data from main memory locations. Those locations are frequently used locations. Cache memories are independent of each other as i-cache, D-cache.

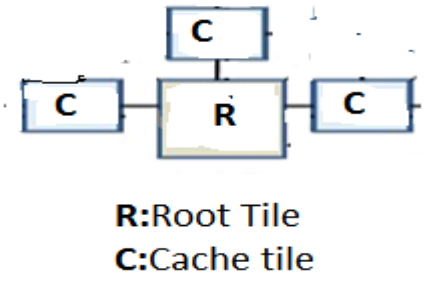

Figure 1:Root tile with level 1 cache

Cache Performance:

Search data in cache would result in either Hit or miss status.
Hit**:** Data in search is there in cache
Miss**:** Data in search is not present in cache**.**

If it experience miss, then data will be taken from main memory
Hit ratio represents the performance of cache memory.

Hit ratio = number of hit / (number of hit + number of miss) = number of hits per total accesses

Cache performance could be improved by using

1.By increasing cache block size,
2.reducing miss rate,
3.reducing time to access data

Cache Mapping:

There are several types of mapping techniques for cache memory from main memory: Direct mapping, Associative mapping, and Set-Associative mapping.

Direct Mapping –

This is simplest mapping technique. Here it maps each data block of RAM into only one possible cache data line. The old block is removed if a line is previously taken up by a memory block, when trying to load new data block. An address space has two field i field and a t field. Direct mapping performance directly depends on the number of hit.

Figure 2: Cache memory address

- Cache memory Address-1)T-Bits 2)I-Bits 3)O-Bits
- O-Bits =$\log_2$(number of words per block)
- I-Bits =$\log_2$(number of blocks in cache)
- T-Bits =address bits-offset bits-index bits
-

2.Background Work

- Kim et al. is first one to introduce Non Uniform Cache Architecture organization[1].He tried to reduce wires between bank and cache controller forming Static-NUCA. This architecture used fixed mapping. It is a Simple design. but no method to bring bank located away from core closer to core.

- Before NUCA the cache architecture was called as UCA consuming latency_41 cycles. Uniform cache architecture shown in fig (a) below.

- S-NUCA was designed (L2) cache, latency was 29 cycles .

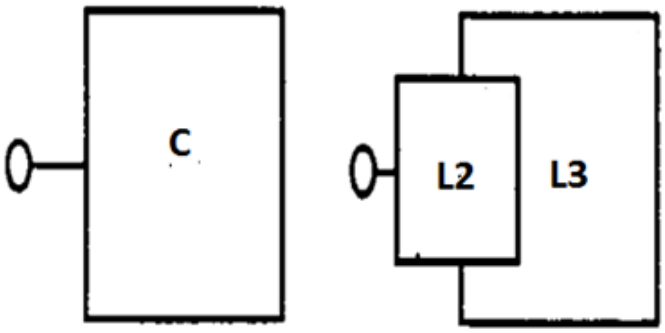

Figure 3: (a)UCA (b)S-NUCA

- D-NUCA allows block migration between the bank[2].Uses mapping is dynamic. Here D-NUCA keep repeatedly used block very close to processor. This dynamic cache management scheme is designed for L2 level of cache in NUCA-based architectures, In this scheme data in cache sharing is performed by adapting promotion migration, insertion policies as per requirements of the different applications with different cache access methods. Here in D-NUCA latency seen was 25 cycles.

- TD-NUCA[3] is designed to reduce power requirement in embedded systems. It is the design as alternative for D-NUCA cach. TD NUCA is Triangular Dynamic-Nuca Cache, It is designed to reduce silicon area of D-Nuca cache. TD-NUCA design reduces the silicon area by 50% compared with Dynamic-NUCA

- Shared Private-NUCA[4] is designed to reduce on-chip latency considering not to affect implementation cost in comparison with conventional static NUCA. Design removes interference between the core. It is able to adjust shared partition considering workload . Each data set in cache memory is portioned into shared banks and private banks. Finally, static and dynamic NUCA and SP-NUCA comparison is done. The results proves that on average shared private NUCA could improve system performance of a dynamic NUCA by around 6%. and static NUCA by 16% .

- This paper presents cache architecture called Enhanced Shared-Private Non-Uniform Cache Architecture (ESP-NUCA)[5], which is suitable for high performance processor. Design is cost effective. Limitation of SP_NUCA is multiple sharers needs to access far away bank to access shared block which increases shared access latency.Replica block is a stored on local partition of cache .It is another copy of shared block.Where a Victim block is on shared partition of cache. It represents private data on remote location.Here in ESP-NUCA above mentioned limitation of SP-NUCA has overcome by by introducing extra group contains of replicas and victims. These are helping blocks. Performance of ESP-NUCA is presented which shows it outperform the private cache by 20% , shared cache by 40%. and D_NUCA by 28%.

- *This paper[6] discuss the performance of several alternatives available for each of available four policies: bank placement, bank access, bank migration and bank replacement. These policies help to determine behavior* .Discussion show us that there is still possibility for improvement in all these policies development.

- This paper presents Dynamic cache management scheme. Which is for Last Level Cache in NUCA-based cache architectures. This adaptive cache management scheme is adaptive. In this design it is possible individual processor cores to access cache capacity from remote location. The size of local partition of last-level-cache is 1 MB with a 10 cycle hit latency. Cache hits in the remote location consume extra latency of 30 cycles.

- This paper[8] implements to use optical NoCs .This design is for cache access protocols for large shared caches. Here level 2 of cache is considered. The problem is unique as optical networks require very low latency. Here all the cache banks are kept closer to each other. Latency_L2_shared_8 cycles.

3.NUCA

- NUCA: Non Uniform Cache Architecture
- Tile: This is a very large distributed cache
- Fig 1 shows a Four Level NUCA. Each Level consist of a group of small cache tiles. First level is the root tile, remaining are large distributed victim tiles. The numbers inside the tile represents latency seen by the processor when the cache hit occurs

Implementation of NUCA involves basic operation search, transport and replacement.

- Search: Search network responsible to send miss request to cache tile.
- Transport: If the cache tile found data to be searched, it transport data to processor.
- Replacement: If the cache tile is not found data to be searched, data needs to be taken from main memory.

Root tile: Root tile is just responsible to forward the request from processor to cache tiles. Here it accepts Request from a processor and forward it to 3 child cache tiles

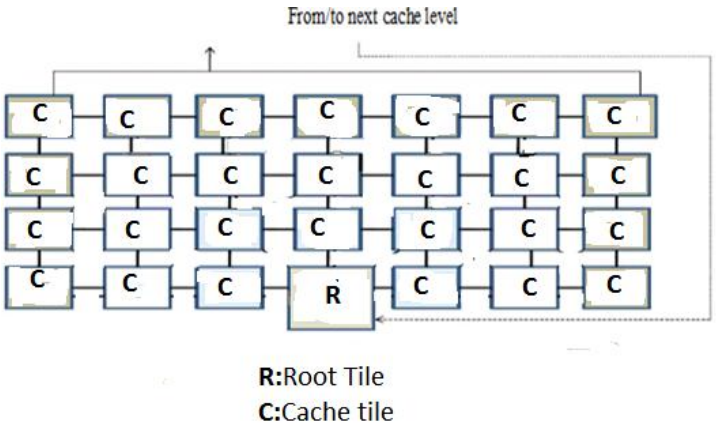

Figure 4: NUCA

4.Result and discussion

Root tile receives request from Processor and forwards the request to three child cache tiles of level1.Req_in is the input trough which it accepts request and addr_out pins for sending request to tiles. Along with request ,address is also sent, which needs to be searched as shown in fig 5.

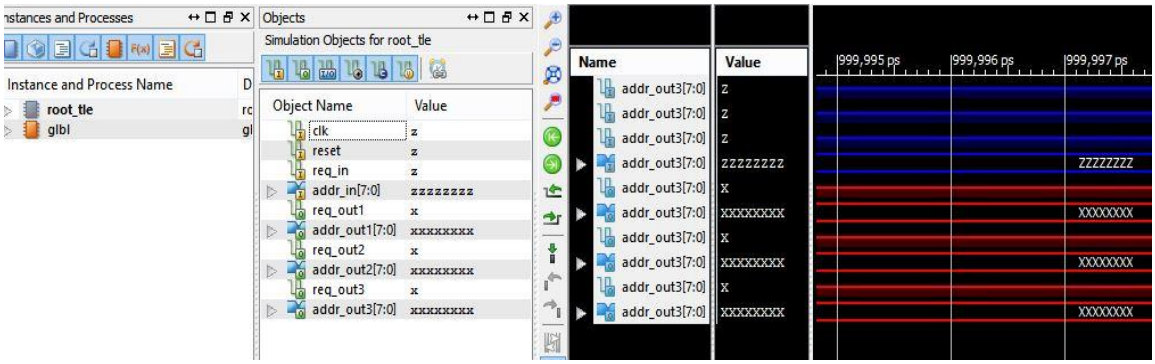

Figure 5: Root Tile

Fig 6 Shown simulation result obtained for design of cache tile. Here cache is of 64 locations. Address is split into tag array, index array.6 bots are required to address 64 locations. Hence tag array is of 6 bits.

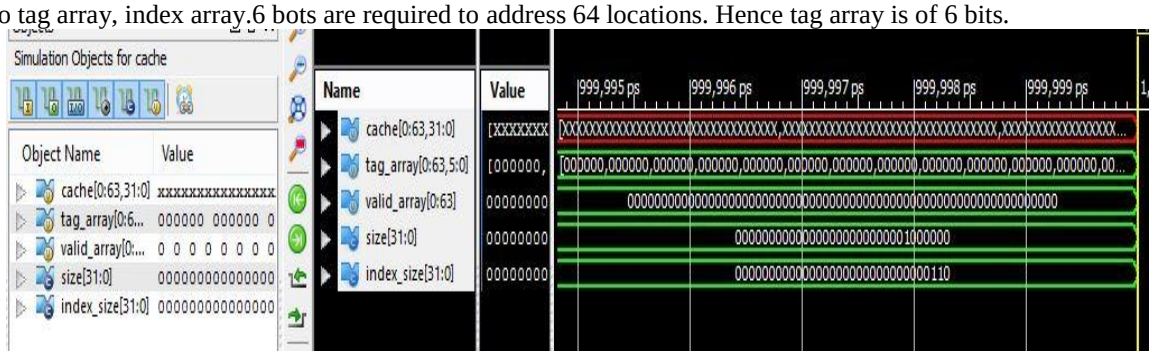

Figure 6: Cache Tile

ACKNOWLEDGEMENT

The authors would like to thank Visvesvaraya Technological University, Belagavi and Department of ECE,

Oxford College of Engineering, Bangalore for the assistance provided for this work.

References

1. C. Kim, D. Burger, and S. W. Keckler, “An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches,” in Proc. 10th Int. Conf. Architectural Support for Programming Languages and Op- erating Systems (ASPLOS’02), 2002, pp. 211–222.
2. P. Gratz, C. Kim, R. McDonald, S. W. Keckler, and D. Burger, “Im- plementation and evaluation of on-chip network architectures,” in Proc. 24th Int. Conf. Comput. Design, 2006, pp. 1625–1628.
3. P. Foglia, D. Mangano, and C. A. Prete, “A nuca model for embedded systems cache design,” in Proc. 3rd Workshop Embedded Syst. Real- Time Multimedia (ESTImedia’05), 2005, pp. 41–46.
4. Javier Merino, Valentín Puente, Pablo Prieto, José Ángel Gregorio" SP-NUCA: A Cost Effective Dynamic Non-Uniform Cache Architecture" in ACM SIGARCH Computer Architecture News, May 2008, Vol. 36, No. 2,
5. Merino, V. Puente, and J. Gregorio, “ESP-NUCA: A low-cost adap- tive non-uniform cache architecture,” in Proc. 16th Int. Symp. High Per- formance Comput. Architecture (HPCA’10), 2010, pp. 1–10.
6. Javier Lira Carlos Molina and Antonio Gonz´alez "Performance Analysis of Non-Uniform Cache Architecture Policies for Chip-Multiprocessors" 28 May 2014.
7. Fazal Hameed, Lars Bauer, and Jörg Henkel "Dynamic Cache Management in Multi-Core Architectures through Run-time Adaptation " 20 May 2014
8. Eldhose Peter, Anuj Arora, Janibul Bashir, Akriti Bagaria and Smruti R. Sarangi", May 2017 Optical Overlay NUCA: A High Speed Substrate for Shared L2 Caches".pp 39-59