Research Journal of Applied Sciences, Engineering and Technology 9(8): 637-644, 2015
DOI:10.19026/rjaset.9.1448
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2015 Maxwell Scientific Publication Corp.
Submitted: October 22, 2014 Accepted: December 18, 2014 Published: March 15, 2015

Research Article

Reconfigurable Architecture for Minimizing the Network Delays in the Multi-core Systems

D. Venkatavara Prasad and Maddineni Deepthi
Department of Computer Science and Engineering, SSN College of Engineering, Chennai, India

Abstract: Noc architecture performs better comparing to bus based when the number of processors is small. On the other hand bus based performs better than noc when number of the processors is large. This leads to new architecture which is hybrid bus based architecture where each node is packet switched in a mesh network of noc architecture that contains bus based system with small number of processors. Few results showed that this hybrid architecture performs optimally better than either purely noc based or purely bus based architecture. Hybrid architecture contains a processor connected to the bus, the bus in turn connected to the router. Each processor contains a private Level 1 (L1) cache. When hybrid architecture is preferable, the optimal number of processors on each bus subsystem varies based on the application. Hence the proposed architecture allows scalable bus-based multiprocessor subsystems on each node in the NoC. This system provides a multi-bus execution environment where each processor is connected to a bus and the bus-based subsystems communicate via routers connected in a mesh-style configuration. The system can be reconfigured to vary the number of bus subsystems and the number of processors on each subsystem. This architecture provides reliability and adaptability and reduces the network delays. Implementing and presenting the details of architecture and experimental results using ns2 indicating the advantages of this architecture.

Keywords: Hybrid architecture, internet protocol, multi-core, network-on-chip, reconfigurable architecture, topology

INTRODUCTION

Network Simulator (NS): Network Simulator (Breslau *et al*., 2000) is a discrete event simulator targeted at networking research. NS provides substantial support for simulation of TCP, routing and multicast protocols over wired and wireless (local and satellite) networks. All of them are discrete-event network simulator, primarily used in research and teaching. The core of NS-2 is also written in C++, but the C++ simulation objects are linked to shadow objects in OTcl and variables can be linked between both language realms. Simulation scripts are written in the OTcl language, an extension of the Tcl scripting language. NS-2 has a animation object known as the Network Animator, nam-1 used for visualization of the simulation output and for (limited) graphical configuration of simulation scenarios (Göktürk, 2006).

NoC architecture: Noc architecture renders the communication infrastructure for the resources (Kumar *et al*., 2002). We have two objectives that why we are going to noc architecture. First create noc by connecting them when the hardware resources can be developed independently. Secondly configurable network is flexible to adapt to different needs of workloads. In noc architecture we are choosing mesh topology as basic topology since it is simple and interconnections are independent of the network between the processors and routers. The advantages of the noc are on chip interconnect bandwidth, cost and ease to use methods to exploit. In noc architecture (Fig. 1) each router has a processor core. The each processor cache has a private l1 cache and second level cache l2 which is shared and distributed among routers. This model of communication is like resources executing the local network computation. Communication between resources in the network is implemented by passing messages. The processor generates memory request and sends it to the router with a packet containing the destination address. The router forwards that request to particular router. Once the request reaches the router, it processes the request and sends it back to source router and router forwards it to its processor. Basic communication between the processors is packet switched which are passing through routers. The memory request process time is measured as the time taken for the memory request to reach the destination plus return back to source processor.

Corresponding Author: D. Venkatavara Prasad, Department of Computer Science and Engineering, SSN College of Engineering, Chennai, India

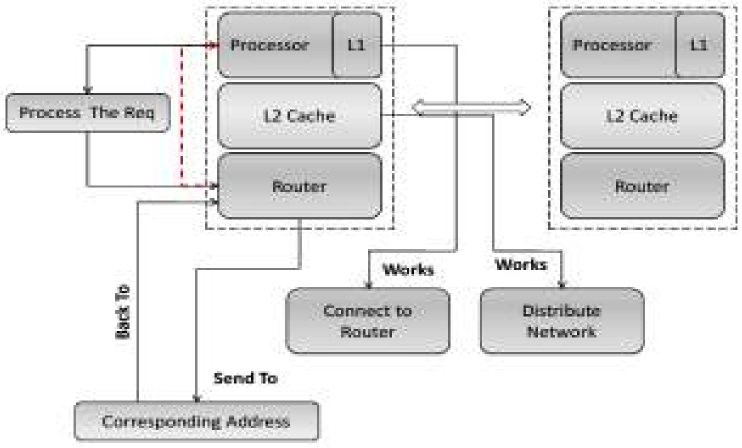


Fig. 1: NoC architecture

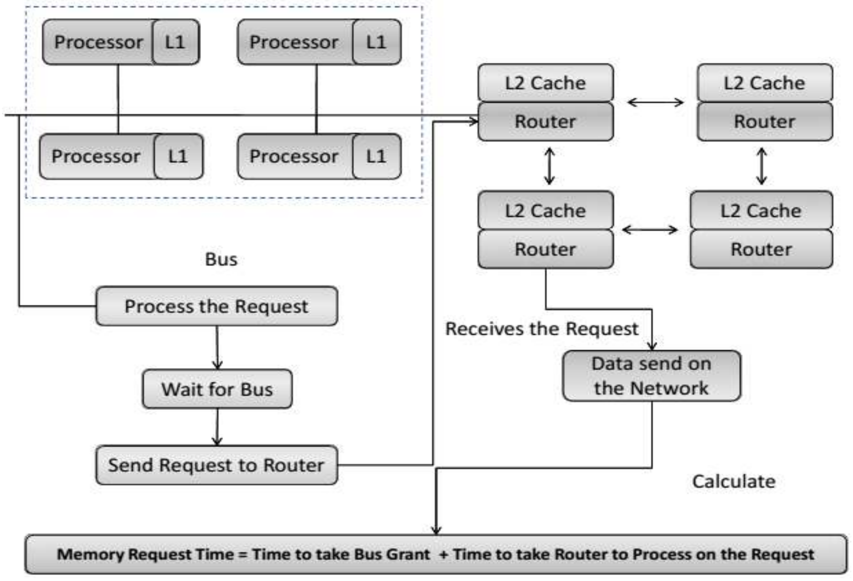


Fig. 2: Hybrid bus based architecture

In this architecture routers can process only one request at a time. The pending requests from different processor cores are queued based on their arrival orders. The delay is also added to overall memory request process time. All links are considered to have uniform and unit delay.

Hybrid architecture: The noc architecture performs better when large numbers of processors are there. In hybrid architecture, (Fig. 2) the processors are connected to bus and where each processor core has a private l1 cache. The bus in turn connected to the router where the router has the shared and distributed l2 cache. Here number the size of the shared cache is p times larger since we are connecting p number of processor cores to the bus (Wang *et al*., 2012). The processor core generates the request and forwards it to the bus. Buses are faster and only smaller area is needed comparing to assigning one router for each processor. This technique reduces the delay and increases the throughput comparing to previous architecture (Venkata Vara Prasad and Maddineni, 2013).

LITERATURE REVIEW

According to Moore's law computing power demands doubles every 18 months. To meet these increasing demand new ideas need to be explored. In tradition method this was met by increasing the individual processors clock speed but it diminishes the returns of the increased speed (Agarwal and Levy, 2007), so new idea of increasing the number of the processor cores in a single chip Simply increasing number of processor cores is too straight and providing connections between them is also a difficult task (Jim *et al*., 2006). Simple solution to this is to use bus topology for communication between the cores. In this design model each processor has its own private l1 cache and also a shared l2 cache among all processors. But this is not viable method because the delay increases when the number of processors increases.

So we are moving to different approaches to reduce the latency, one of such approach is noc. In noc each processor is connected to router and communication between them is done using protocols. Each processor has its own private l1 cache and a shared and distributed l2 cache among routers. Latency to obtain data may differ depending on the number of hops. If the l2 cache closer then few hops are required to obtain the data. For reducing the latency in noc architecture sub mesh interconnects was introduced. In noc 2D mesh topology is the popularly used topology. It has mxn tiles where each is connected to four neighbors. Since it is connected to neighbors it suffers from large hop counts so other topology also used for noc like flat or hierarchical. Hierarchical architectures have small number of hop counts for global packets but bandwidth is reduced. Hybrid architecture uses mesh topology for local routing and hierarchical interconnects for global routing. It splits the larger mesh into smaller which are connected by hierarchical interconnect for global traffic (Seungju *et al*., 2012). Traffic at the center is reduced by routing the traffic to the corners of the mesh but results in reduced latencies. By placing resources nearer to bridge station increases latencies. By placing the bridge station away from corner still more increases the performance. If there is increase in mesh size again it leads to increase in size of sub-mesh so it is recommended to fix the sub mesh size and increasing levels in the hierarchical (Bourduas and Zilic, 2007).

PROPOSED ARCHITECTURE

The noc architecture performs better when large number of processors is there. But when smaller numbers of processors are they this architecture is less likelihood to have because of its unit delay in the mesh topology. In the case of bus, it performs better than noc for smaller number of processors. But when the number of processors cores is large in number the bus saturates earlier. This leads to new architecture called hybrid bus based noc architecture where both the bus and mesh styled noc architectures are combined (Reddy *et al*., 2014).

In hybrid architecture (Avakian *et al*., 2010), the processor cores are connected to bus, where each core has a private l1 cache. The bus in turn connected to the router where the router has the shared and distributed l2 cache. Here number the size of the shared cache is p times larger since we are connecting p number of processor cores to the bus. The processor core generates the request along with the packet containing destination router address and forwards it to the bus. The bus need to grant the memory request. Once the bus grants the request, it forwards the request to the router. After router receives the request, it processes the request an switched when it need to be send to different router and sent over network. The memory request process time is d sends it back to source. The memory request is packet sum of the time taken for bus grant, time to reach the destination and time to return back to source. Buses are faster and only smaller area is needed comparing to assigning one router for each processor. The disadvantage of hybrid bus based noc is that the assigning of processor cores for an application is static so moving to another architecture called reconfigurable architecture.

In reconfigurable architecture (Fig. 3) (Avakian *et al*., 2010) each processor core has a local L1 cache and the L2 cache is shared and distributed equally among the routers. The routers are connected as a mesh.

Communication between routers is packet switched (Grover *et al*., 2011). The processor cores are also connected as a mesh. But each segment of the bus is controlled by a switch. Based on the configuration of the switches, buses can be created dynamically. Once the buses are formed, they need to be connected to the

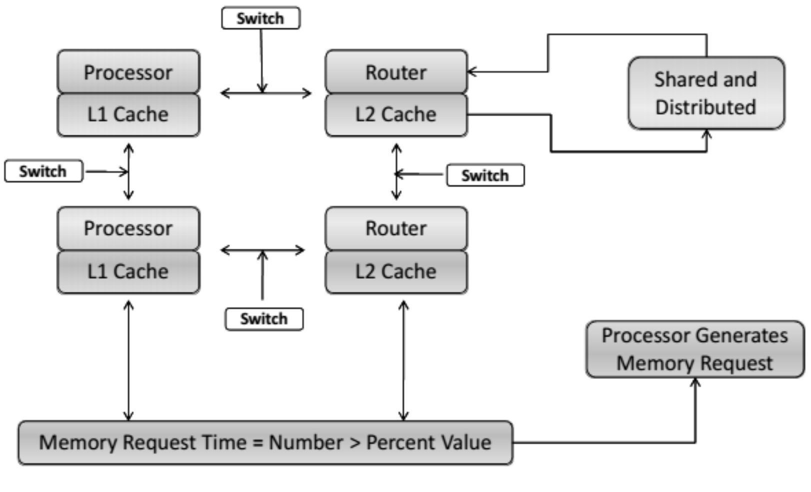


Fig. 3: Reconfigurable architecture

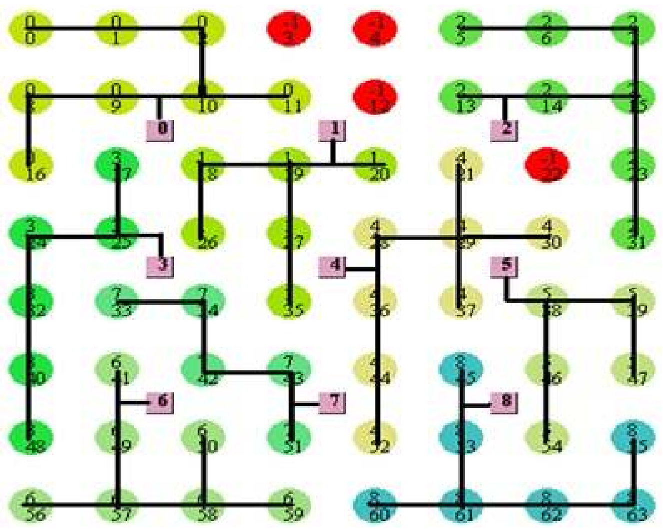


Fig. 4: Examples for mapped cores

routers. Each router has 4 access points to the bus. Therefore the architecture provides hybrid bus network model. It also gives the flexibility of connecting as many processor cores on a bus as needed. Adaption of core failure is also simple. The OS can disregard the dead processor core and remove it from the available cores list (Avakian *et al*., 2010). The switches need to be configured so as to form a bus and connect the bus to a router to form a hybrid bus-NoC instance. In order to map cores to routers, several constraints need to be taken into consideration while configuring the system. Listed below are some of those constraints:

- No processor should be connected to more than one router at the same time
- No core should be stuck isolated and unusable
- All cores need to be shared equally between routers on a need basis
- Any mapping of cores to routers should be possible

In order to meet this constraint, the proposed architecture assigns priority to each core. The priority values start at p which is equal to the number of cores per row. The cores on the boundary have the highest probability of getting stuck, therefore the priority of those corner cores are set to the highest. Following that, the priority decreases while going inward as they do not have a high probability of being stuck. The proposed architecture first assigns priorities to each processor core. When a process asks for NP (Needed Processors) cores, the architecture lists the routers which have at least NP cores available. It then chooses the router that has the least amount of available cores that is greater than NP. Once the router is chosen, the architecture then starts forming the bus. It enqueues the available neighboring cores based on their priorities. The processor core connected to the router is the first core in the queue. It then starts queuing the neighbors of the cores in the priority queue. It continues this process until NP cores are used to form a bus.

Each core has a number on the top which represents that the particular processor core is assigned to that router (Zhang *et al*., 2011) as in the Fig. 4. If the number is -1 means then that particular core is not assigned. The proposed architecture guarantees an assignment if a path exists since it checks the number of available cores for every router. If a hardware failure should occur, then the architecture can assume the failed processor core is always assigned and continues operation without major disruption.

Simulations are run for the three architectures noc, hybrid and reconfigurable architecture. The delay time and throughput are calculated for generated memory requests. Delay time includes the bus grant time, waiting time in router queues and request process time. We considered unit delay for all unit links.

EXPERIMENTAL RESULTS

Wired topology is created and stimulated three architectures in ns2 (Breslau *et al*., 2000). Memory requests are generated and send to destination router; it processes the request and sends the results back to the source.

Delay time and throughput are the two parameters estimated for these three architectures for analyzing their performance. First implement the noc architecture (Fig. 5) in ns2 with processor and private l1 cache and routers with shared cache. Memory requests are generated and sent to router processes it and sends the results back. Delay time s calculated for processing these requests in this architecture.

Second implement the hybrid bus based (Fig. 6), generating memory requests and sending it to the bus. Once the bus grants the requests sends it to router and router processes the requests. Delay time and throughput is estimated for this architecture.

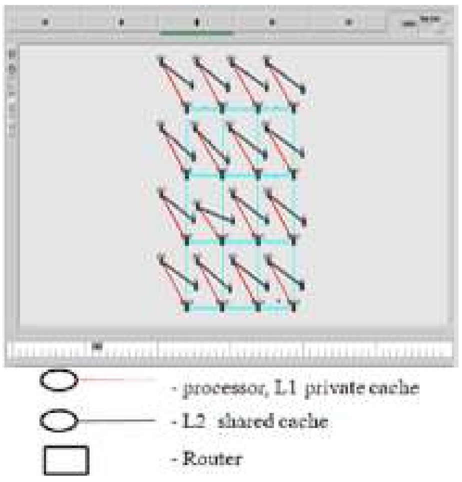


Fig. 5: Topology of NoC architecture

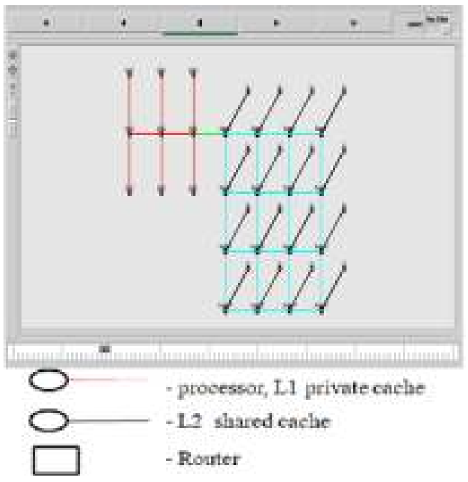
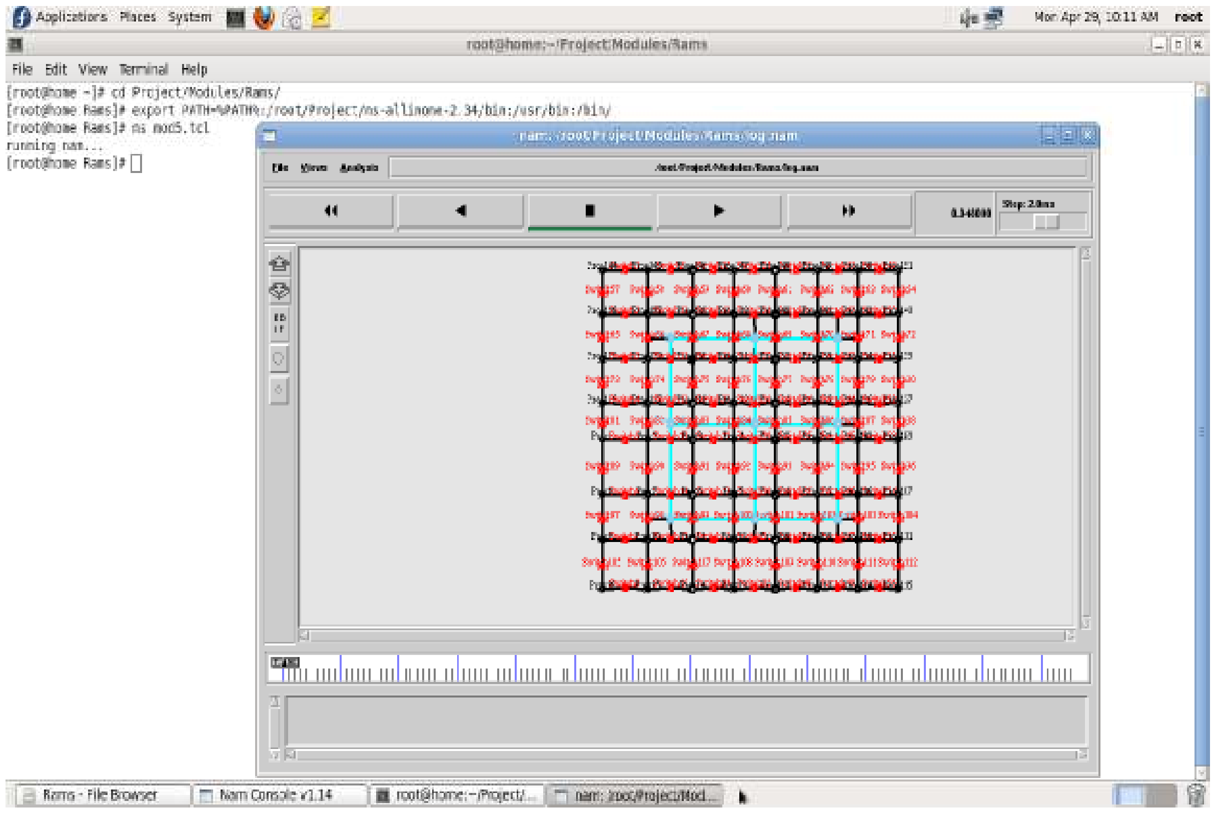


Fig. 6: Topology of hybrid bus based architecture


Applications Places System
Mon Apr 29, 10:11 AM root
root@home:~/Project/Modules/Rams
File Edit View Terminal Help
nam: /root/Project/Modules/Rams/og.nam
File Views Analysis
Rams - File Browser
Nam Console v1.14
root@home:~/Project/...
nam: /root/Project/Mod...

```
[root@home ~]# cd Project/Modules/Rams/
[root@home Rams]# export PATH=$PATH:/root/Project/ns-allinone-2.34/bin:/usr/bin:/bin/
[root@home Rams]# ns mod5.tcl
running nam...
[root@home Rams]# 
```

Fig. 7: Topology of reconfigurable architecture

Thirdly implemented the reconfigurable architecture (Fig. 7 and 8) and generated the memory requests, Assigning the priorities to the processor cores and sending it to the routers based on their priorities. Once the router receives the memory requests it processes the requests. Delay time and throughput is estimated for this architecture.

The performance analysis of these three architectures is shown in the Fig. 9 clearly tells that the delay is reduced in the reconfigurable architecture comparing to noc and hybrid architecture. Figure 10 shows the throughput of reconfigurable architecture is improved when comparing to noc and hybrid architecture.

Figure 11 shows the delay time and throughput estimated for different number of memory requests for these two noc and hybrid architecture. Delay in hybrid architecture was reduced by 4% when comparing to noc architecture. Throughput of hybrid is improved by 17% comparing to noc architecture.

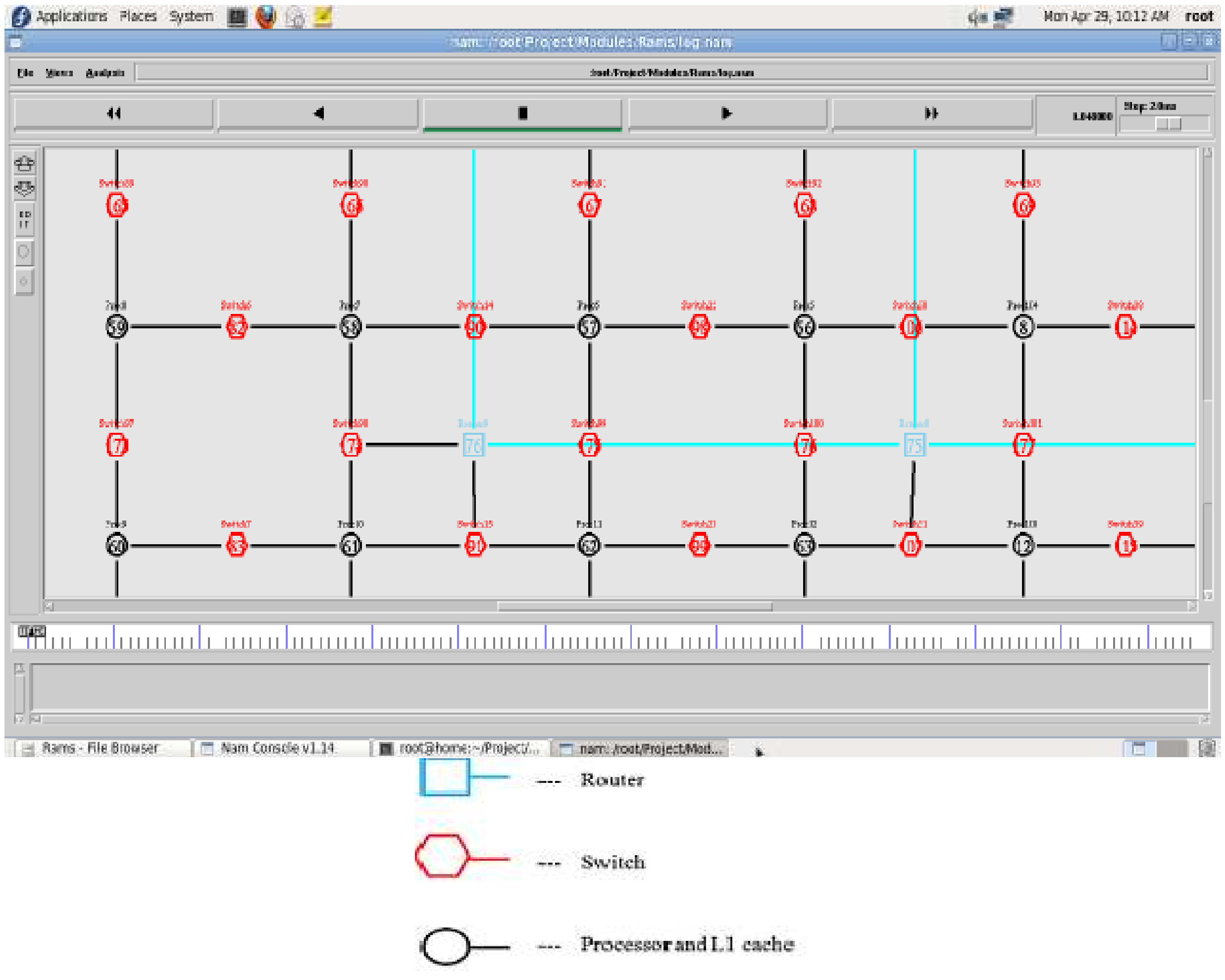


Fig. 8: Zoomed view of reconfigurable architecture

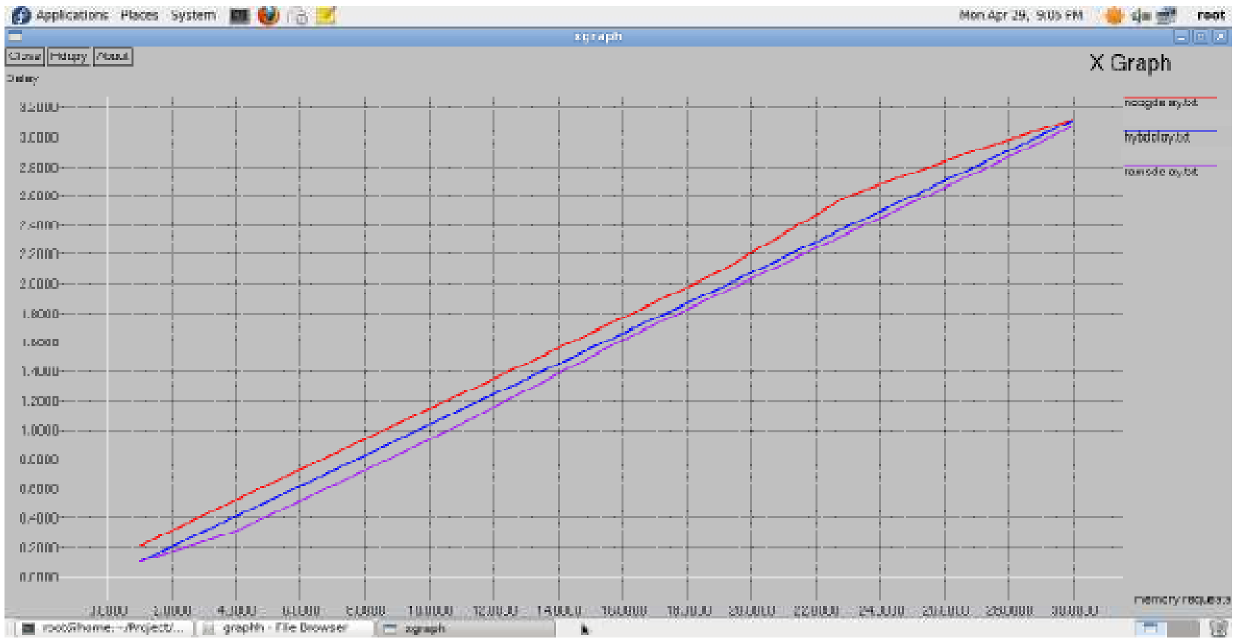


Fig. 9: Graph comparison for delay

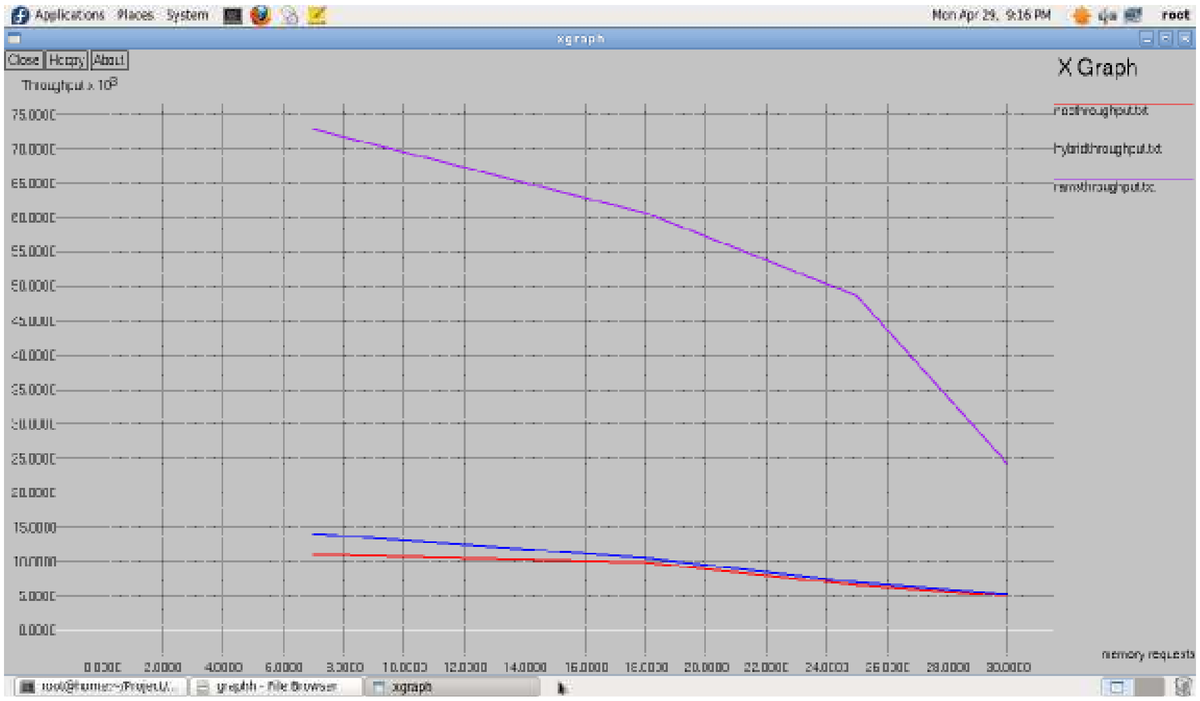


Fig. 10: Graph comparison for throughput

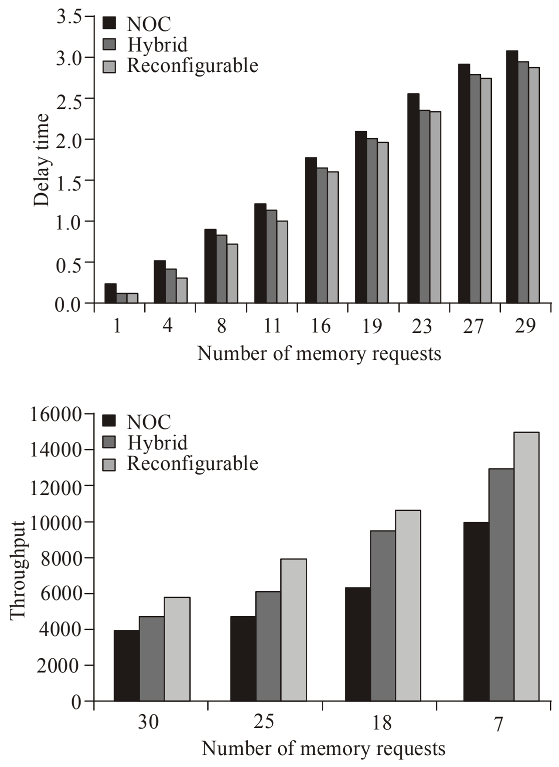


Fig. 11: Bar chart for delay and throughput comparison

CONCLUSION

NoC and Hybrid Bus Based architecture are important to obtain performance benefits of the both. The advantages of bus for smaller number of processors can be combined with the advantages of noc for larger number of processors to reduce the delay and to increase the processor utilization of the architecture. Performance of hybrid architecture is improved comparing to NoC architecture. Finally implementing the Reconfigurable architecture by configuring the processors to generate memory requests and sent those requests to the routers based on priorities assigned to the processors to reduce the delay and to provide optimized processor utilization in the architecture. The performance of three architectures is compared and delay time is reduced and processor utilization is optimized in the reconfigurable architecture comparing to NoC and Hybrid.

REFERENCES

Agarwal, A. and M. Levy, 2007. The kill rule for multicore. Proceeding of the 44th ACM/IEEE Design Automation Conference (DAC'07), pp: 750-753.
Avakian, A., J. Nafziger, A. Panda and R. Vemuri, 2010. A reconfigurable architecture for multicore systems. Proceeding of the IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW, 2010), pp: 1-8.
Bourduas, S. and Z. Zilic, 2007. A hybrid ring/mesh interconnect for network-on-chip using hierarchical rings for global routing. Proceeding of the 1st International Symposium on Networks on-Chip (NOCS'07), pp: 195-204.

Breslau, L., D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy *et al*., 2000. Advances in network simulation. Computers, 33(5): 59-67.

Göktürk, E., 2006. Design of a higher level architecture for network simulators. Proceedings of the 20th European Conference on Modelling and Simulation (ECMS, 2006).

Grover, S., A. Dhanotia and G.T. Byrd, 2011. A canonical multicore architecture for network routers. Proceeding of the ACM/IEEE 7th Symposium on Architectures for Networking and Communications Systems, pp: 134-144.

Jim, H., B. Jerry and K. Sean, 2006. From a Few Cores to Many: A Tera-scale Computing Research Overview. White Paper at Intel 2006, (Online).

Kumar, S., A. Jantsch, J.P. Soininen, M. Forsell, M. Millberg *et al*., 2002. A network on chip architecture and design methodology. Proceeding of the IEEE Computer Society Annual Symposium on VLSI, pp: 105-112.

Reddy, T.N.K., A.K. Swain, J.K. Singh and K.K. Mahapatra, 2014. Performance assessment of different network-on-chip topologies. Proceeding of the International Conference on Devices, Circuits and Systems (ICDCS'14). Coimbatore, pp: 1-5.

Seungju, L., Y. Masao and T. Nozomu, 2012. A locality-aware hybrid NoC configuration algorithm utilizing the communication volume among IP cores. IEICE T. Fund. Electr., E95-A(9): 1538-1549.

Venkata Vara Prasad, D. and D. Maddineni, 2013. Optimizing the processor utilization of the hybrid architecture in multicore systems. Proceeding of the International Conference on Computational Intelligence and Engineering Applications, Interscience Research Network.

Wang, Y., S. Chen, K. Zhang, H. Chen and X. Chen, 2012. Architecture design trade-offs among VLIW SIMD and multi-core schemes. Proceeding of the IEEE 26th International Parallel and Distributed Processing Symposium Workshops and PhD Forum (IPDPSW), pp: 1649-1658.

Zhang, L.L., L. Dong-Sheng and Y. Ai-Xia, 2011. A static routing algorithm based on the 3D mesh structure [J]. Comput. Eng. Sci., 11(2011): 016.