STUDY ON APPLICATION AWARE PRIORITIZATION ON NETWORK ON CHIP

[1]Sarath Babu,[2]Sangeetha Jose

[1]MTech Scholar,[2]Assistant Professor

[1]Network Engineering

[1]Government Engineering College Idukki, Kerala

Abstract :The emerging fabrication technologies allow the integration of many processor or processing elements on a chip to form a multiprocessor. Network-On-Chip (NoC) is an emerging technology that interconnect resources (processing elements and routers) inside a multicore processor chip. Communication subsystem connects processing elements to a network that routes packets between them. Compared to the traditional bus interconnection system, Network-on-chip provides scalable bandwidth and power. Traditional Network-on-Chips (NoCs) employ simple arbitration strategies, such as round robin or oldest-first, to decide which packets should be prioritized in the network. Major challenges of these scheduling policies are equal prioritization of the packets and their application-oblivious nature. Hence different packets can have different effects on system performance due to the memory level parallelism(MLP).

Applications can be network latency sensitive and latency tolerant. Application can be classified and can be prioritized based on the criticality of the packet. This increases the research scope of application aware routing. However, major challenge is to develop policies and mechanisms that enable multiple diverse applications to efficiently share the network and hence improve the performance. Instead of application oblivious routing, application aware routing can improve system level throughput and performance. This paper focuses on different prioritization techniques such as age based, round robin based, Stall-Time Criticality based and slack based prioritization. By comparing those different schemes, it is clear that STC and slack based have better performance result. While changing the slack to a dynamic property by updating slack value on every router on its traversal, the performance increases.

***IndexTerms* - Memory Level Parallelism, Network-on Chip, Stall-Time Criticality**

I. INTRODUCTION

Processor is the complex circuit that can process instructions of a particular task. The operations are fetch, decode, execute and write back. Figure 1 shows one type of processor called unicore processor, which has single core inside a chip, running a single thread at any one time. Figure 2 shows the another type of the processor known as Multicore processor. The modern fabrication technology allows integration of multiple independent processor inside a chip. Now a days, multicore processors are used commonly. Single processor can run multiple instructions on separate cores at the same time, which increases overall speed for programs and enable to parallel computing. Manufacturers typically integrate the cores onto a single integrated circuit die, or onto multiple dies in a single chip package. Multi-core processors are widely used across many application domains, including general purpose, embedded, network, digital signal processing(DSP), and graphics (GPU).

Multicore processors require a communication subsystem for interconnecting the resources for implementing message passing or shared-memory inter-core communication methods. The common network topologies to interconnect cores inside a chip include bus, ring, two-dimensional mesh, and crossbar. The performance of most digital systems in the current era is limited by their communication or interconnection, not by their logic or memory. Memories and processors become small, fast, and inexpensive. The pin density and wiring densitythat govern interconnections between system components are scaling at a slower rate than the components themselves. Traditional processors using bus as interconnection system. Buses having many limitations like not scalable, bandwidth limitations, central arbitration bottleneck and long wire delay. The problems affect the processor throughput and performance. Hence require a more efficient communication pattern known as NoC (Network on Chip) [1].

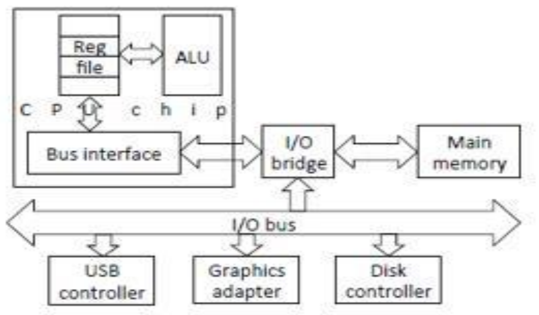


Figure 1. Unicore Processor

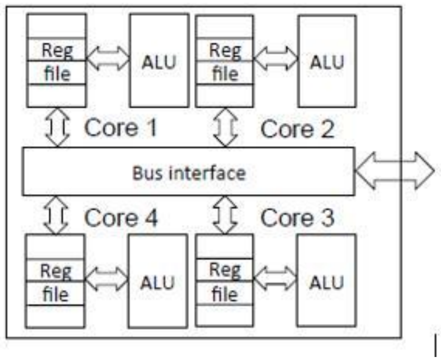


Figure 2. Multicore Processor

II. RELATED WORK

2.1 Components of NoC

NoC consists of processing elements (PEs), network interfaces (NIs), and routers. Figure 3 shows the conceptual diagram of Network on Chip. The modules are described below:

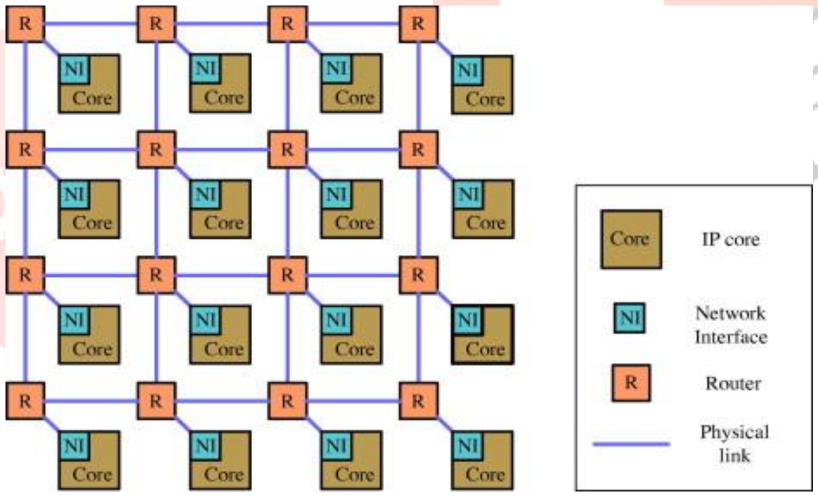


Figure 3. NoC based communication system

1. Processing Element(PE): This includes processors, memories and dedicated hardware. All these processing elements can be arranged in different topologies for efficient communication and optimum performance.
2. Network Interface (NI): The Network Interface is used to packetize data before using the router backbone to transverse the NoC. Each PE is attached to an NIC which connects the PE to a local router. The network interface is the essential component of NoC architecture. The communication between switch and resource is carried out through Network Interface. Proper communication mechanism along with a well-defined network interface forms the backbone for NoC. This concrete backbone network can be easily established just by putting resources at the interfaces. Such type of design practice improves the reusability and modularity of the network. Network interface can be implemented at both software and hardware levels.

3. Router: Router plays an important role in NoC. Router decides the path of packet travel from source to destination

2.2 Architecture of Router

When a packet was sent from a source PE to a destination PE, the packet is forwarded hop by hop on the network via the decision made by each router. For each router, the packet is first received and stored at an input buffer. Then the control logic in the router are responsible to make routing decision and channel arbitration. Finally, the granted packet will traverse through a crossbar to the next router, and the process repeats until the packet arrives at its destination.

Figure 4 depicts architecture of a router. The NoC Router [4], [13], [15] consists of three major blocks such as crossbar, buffer, and arbiter with five input ports and output ports. The ports are local, east, west, north and south. Packet transmission is done on these five input ports, from source to destination. There are often multiple input buffers for each physical channel so that flits can flow as if there are multiple "virtual" channels. Each input port contains buffers for storing incoming flits and it is logically divided into virtual channels.

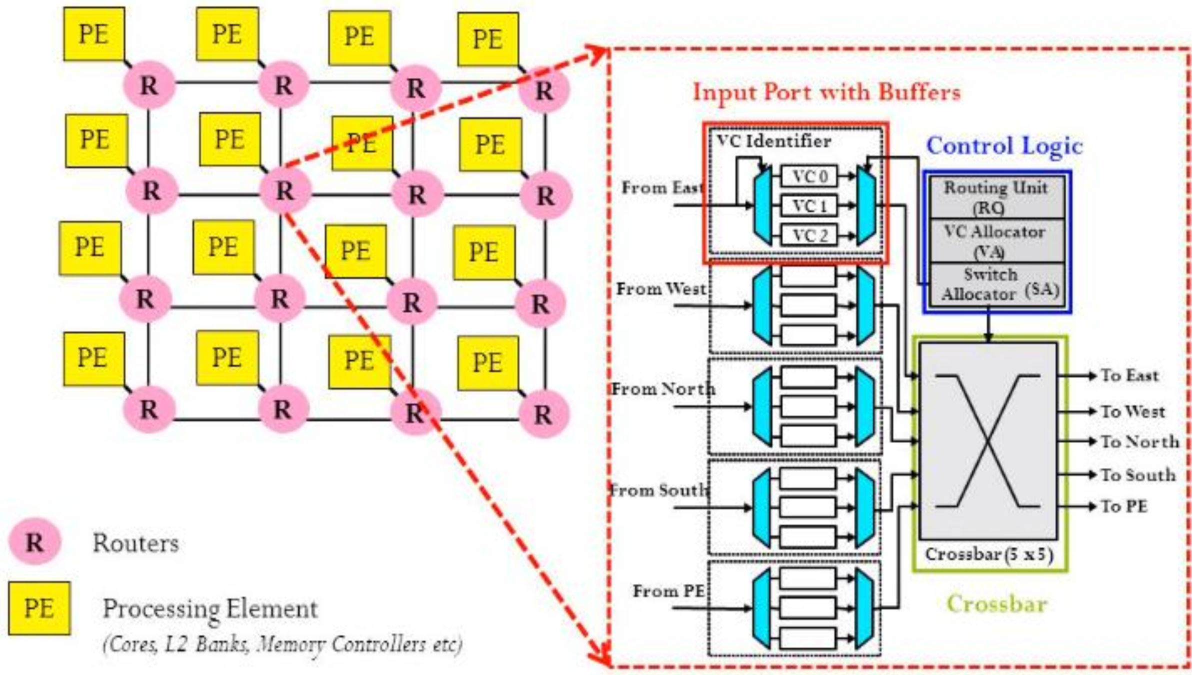


Figure 4. Router Architecture

There is crossbar switch which is controlled by a control logic. The control logic contains three major control modules: a router, a virtual channel(VC) allocator, and a switch allocator. The routing operation takes in four steps or phases such as routing(RC), virtual-channel allocation(VA), switch allocation(SA), and switch traversal(ST). Which often represent one to four pipeline stages in modern virtual channel routers. When a head flit (the first flit of a packet) arrives at an input channel, the router stores the flit in the buffer for the allocated virtual channel and determines the next hop for the packet (RC phase). Given the next hop, the router then allocates a virtual channel in the next hop (VA phase). Finally, the flit competes for a switch (SA phase), if the next hop can accept the flit, and moves to the output port (ST phase), which often represent one to four pipeline stages in modern virtual-channel routers. When a head flit (the first flit of a packet) arrives at an input channel, the router stores the flit in the buffer for the allocated virtual channel and determines the next hop for the packet (RC phase). Given the next hop, the router then allocates a virtual channel in the next hop (VA phase). Finally, the flit competes for a switch (SA phase) if the next hop can accept the flit, and moves to the output port (ST phase). When packets reached in virtual buffer, arbiter wants to pass the packets from input buffer to output port. This process is known as packet scheduling. Packet scheduling is done by the VC and SA modules in router. NoC routers are connected to each other via links(L), which are repeated copper interconnects. The routers are pipelined structures with basic stages [15] described below.

1. Buffering: During this stage a flit is stored inside the input buffers.

2. Route Computation: The destination of the flit is deter-mined from the header flit. Route computation stage then decides the best route that the flit should take to reach its destination. This stage implements a routing algorithm that decides the best route.
3. Switch Arbitration: The flit arbitrates for the available output links during this stage.
4. Switch Traversal: After the output link allocation, the flit traverses the crossbar to reach the output buffers corresponding to the output link.
5. Link Traversal: The final stage of the pipeline is the link traversal where the flit traverses the output link towards the destination router.
6. Routing Algorithms: Routing algorithms [5] are used to decide the best path that a flit will take to reach its destination router. As the performance and power consumption of a network is dependent on the paths that flits take (shortest or longest), routing algorithms play an important role in controlling the power-performance characteristics.
   a. Oblivious Routing: In oblivious routing, the path is completely determined by the source and destination address. This type of routing enables simple and faster router designs. However, oblivious routing algorithms are not able to address various runtime issues such as congestion.
   b. Adaptive Routing: In adaptive routing, the path is decided dynamically based on the runtime conditions such as network congestion. This way, they are able to achieve better performance as compared to the oblivious routing algorithms. However, the downside is a more complex and costly adaptive router.

2.3 Study on prioritization policies

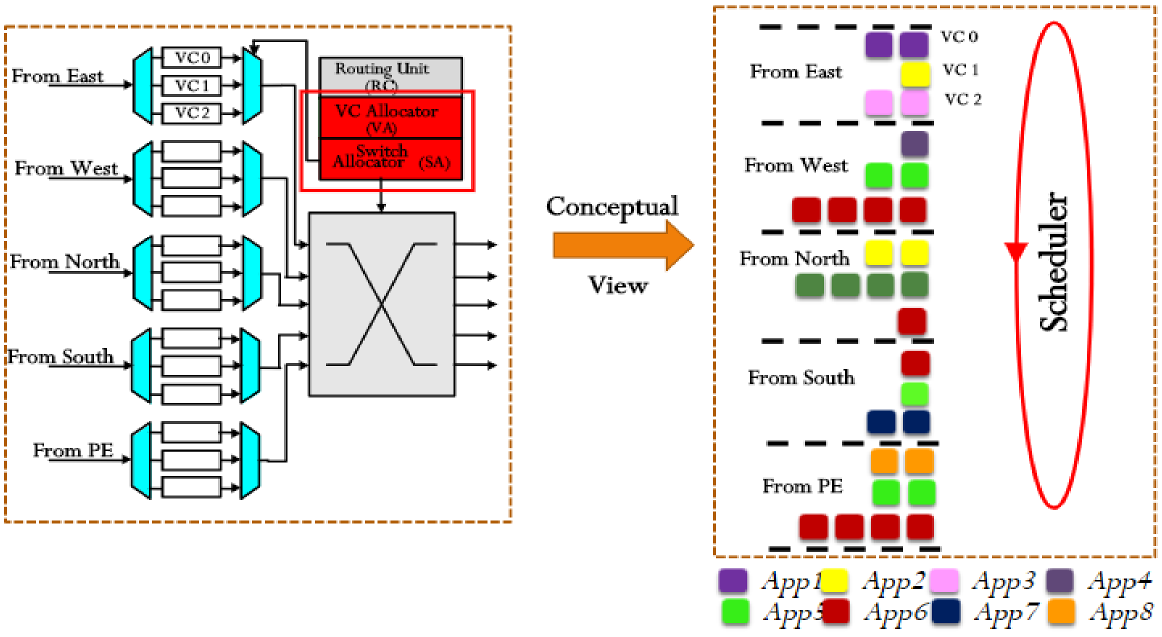


Figure 5. Packet scheduling

Figure 5 shows conceptual view on packet scheduling with a packet scheduler. When all the input ports are request for the same output port contention is occur, to avoid this contention arbiter are used in NoC Router.

In NoC Router different arbitration techniques are used. Traditionally age based and round robin based arbitration policies are used.

1. Age based prioritization: Age based approach [6], [7] is similar to a queue. It is a First Come First Serve model. Prioritization based on the age of the packet, by extracting the age from flit. The aging is local to a particular router. Arbitration is latency fair and oldest requester for a resource is always served first. When multiple packets are competing for a resource, the oldest, measured as the time since its injection into the network, gets access first. Total delay is the

sum of queueing delay and routing delay. Packets are not get any queueing delay when age rate is too slow. Packets ages will saturate very quickly perhaps after only a few hops, when age rate is too fast. While age-based arbitration greatly improves the latency fairness of networks, it is generally used only as a local approximation to a latency fair network. One of the major problem is that the information is local to the router. Packets from one application may be prioritized at one router, to be delayed at next.

2. Round robin based prioritization: A round-robin arbiter [8] operates on the principle that a request that was just served should have the lowest priority on the next round of arbitration. Arbiter controls the arbitration of the ports and resolves contention problem. It keeps the updated status of all the ports and knows which ports are free and which ports are communicating with each other. Packets with the same priority and destined for the same output port are scheduled with a Round-Robin Arbiter. Suppose, in a given period of time, there was many input ports request the same output or resource, the arbiter is in charge of processing the priorities among many different request inputs. The arbiter will release the output port which is connected to the crossbar once the last packet has finished transmission. So that other waiting packets could use the output by the arbitration of arbiter. A round-robin arbiter operates on the principle that a request which was just served should have the lowest priority on the next round of arbitration.

2.4 Globally-Synchronized Frames(GSF)

The GSF framework [9] can be easily integrated with a conventional NoC router. It provides a deadline based arbitration scheme which guarantees bandwidth other than the current time according to a deadline assignment policy. When multiple packets competing for a single path, GSF provides prioritization mechanisms. The priorities of packets are compared to determine which packet is allocated buffer space and switching bandwidth first. This technique guarantees on minimum bandwidth and minimum network delay. Each application experiences and achieves equal network throughput. Frame-based approach groups a fixed number of time slots into a frame. This is similar to time based batching concept. Figure 6 represents quantization of time into equal sized frames.

Each source node can inject an equal and fixed quota of flits into each frame. Once a source node fills its quota in a frame, it then injects packets into the next frame. Each packet is assigned with a global frame number. Packets belonging to the older frames are prioritized over the younger frames. There is no prioritization over packets belonging to the same frame. This global frame numbering system can eliminate expensive book keeping logic and storage at each node.

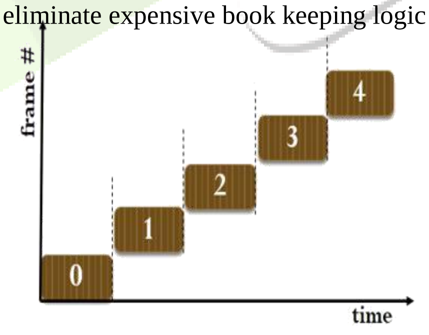


Figure 6. Framing in GSF

2.4.1 GSF Router Architecture

GSF Router architecture includes both carpool lane buffer sharing and early frame reclamation along with the baseline GSF. Starting from a baseline VC router, there are various aspects and design issues in the GSF router. A three stage pipelined VC router is used by the GSF router. Switching bandwidth and buffer allocation determines a sequence of frame buffers to be used by the packet along the path. There can be contention between packets within a frame but not across frames. Allocating switching bandwidth will give highest priority to the earliest frame in the window.

Each router can inject a certain number of flits into each active frame. Conventional frame based bandwidth allocation schemes having some similarity with this concept. Once a packet injected, it traverses the network using only the allocated frame buffers for its given frame number. Within a frame there is no priority for individual flits. Within the frame, here using earliest-frame-first scheduling for bandwidth allocation. Packet contention be occur within the frame but not across frames. If packet from the earliest frame were exist in the network, then that cores are prevented from injecting new packet in the next frame.

Network Stall Time(NST) is the number of cycles the processor stalls and waiting for network transactions to complete. Hence stall time is taken to distinguish the packets. Memory-Level Parallelism, Memory Access Latency, and Network Load are the factors affecting the criticality of the packet.

2.5 Application Aware Approach

The basic idea is to divide processor execution time into phases and rank applications within a phase. Two principles used to enable prioritization are application ranking and packet batching. These ranks are global to the network, hence all routers in the network prioritize packets based on that applications ranks. For ranking the applications, distinguishes applications based on the stall-time criticality (STC) [11] of their packets. Since applications are heterogeneous, different applications having different criticality [10] with respect to the network. Some of them are network latency sensitive and some of them are network latency tolerant.

1. Application Ranking: In the case of a multicore sys-tem, multiple packets running on multiples cores and these different packets can have different criticality with respect to the network. In Order to prioritize these packets, the packets are first distinguished based on the stall time criticality, then rank the application based on the criticality of the packet. Different factors affecting stall time criticality of the packet are Memory-Level Parallelism(MLP), Memory Access Latency and Network Load.
2. Packet Batching: Prioritization of higher rank applications may cause starvation of lower ranked application. Batching can be used to avoid the starvation by grouping the cycles from first T cycles. At the end of a batch, new batch will have started. Older batches get higher priority than younger batches.

2.6 Slack based Approach

Aergia [12] discusses about prioritization of performance critical packets based on their slack value during packet scheduling. In this scheduling mechanism, slack value of a packet is the number of cycles the packet can be delayed without affecting the execution of the application. In this prioritization mechanism, a critical packet with low slack value will be prioritized over the packet with high slack value. Higher slack valued packet may be starved due to the prioritization of lower slack valued packet. It can be avoided by introducing a batching mechanism.

Slack can be defined [14] as in two different ways. They are:

1. Local slack: It is the number of cycles a packet can be delayed without delaying any subsequent instruction.
2. Global slack: It is the number of cycles a packet can be delayed without delaying the last instruction in the program.

Slack value of packet can be computed as follows:

Slack of packet = Maximum predecessor latency -Latency of current packet(1)

Due to the difficulty of predicting network latency in a realistic system, here categorizes slack into different priority levels based on indirect metrics. The factors affecting criticality of a packet are:

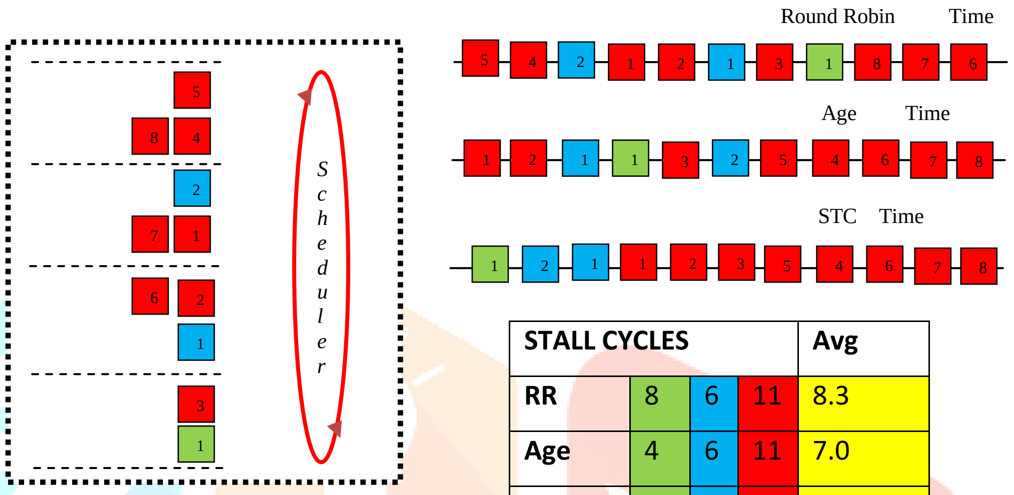
1. Number of predecessor with L2 cache miss
2. L2 cache hit /miss status
3. Slack of the packet

This technique having two principles for prioritization. They are:

1. Slack based prioritization: Lower slack valued packets are prioritized over the higher slack valued packet during arbitration. Slack value is carried out only the head flit.
2. Packet batching: Prioritization of lower slack valued packet may cause starvation of higher slack valued packet. Hence here using a batching mechanism. In which, time is divided into equal and fixed sized (T cycles) intervals. This is known as time based batching.Packet belong to the older batch are prioritized over the packet belong to the younger batch.

III. COMPARISON

This section examines the amount of time taken by routers to service all packets of each application using different prioritization policies. Examine the three comparison mechanisms named Round robin, Age, and Stall Time Criticality.



STALL CYCLES				Avg
RR	8	6	11	8.3
Age	4	6	11	7.0
STC	1	3	11	5.0

Figure 7. Scheduling patterns of different policies

In figure 7, the red packet shows the packet injected from core A, blue packet shows the packet injected from core B and green packet shows the packet injected from core C. Figure 7 shows the difference in these approaches, when the packets are arrived into the virtual channel of a router having 8 VCs. In the case of local Round Robin (local RR), it gives equal priorities to all in the virtual channel. Hence, the green packet will be serviced after all other virtual channels are serviced once. Then the green packet will be serviced at the 8th cycle. The red packet service completed at 11th cycle and the blue packet service completed at 6th cycle. The average application stall time for local Round Robin arbiter is 8.33 cycles.

In the case of local age based prioritization scheme, prioritize the oldest packet first. Hence green packet scheduled in the 4th cycle. It is one of the oldest packet. The blue packet service completed at 6th cycle, red packet serviced at 11th cycle. The average application stall time for local age arbiter is 7 cycles.

In the case of Stall Time Criticality(STC) based prioritization scheme, shortest jobs are prioritized first. Hence the green packet will be serviced first, then it gives priority to the blue packet and last it prioritizes red packet. The average application stall time for STC arbiter is 5 cycles. From that, it can be observed that the performance of STC is better than other two schemes.

GFS aim to improve the quality of service. GFS guarantees on minimum bandwidth, minimum network delay and equal throughput for each application. It is clear that the aim of GFS is different from other two approaches (Age and Round Robin). Hence it gives a concept batching mechanism that can improve the QoS. It is adopted in the modern technique. The Main disadvantage of GFS is that; it does not provide prioritization of packet within the frame.

Slack based prioritization mechanism provides packet prioritization based on the criticality of the packet. The problem is that it computes slack dynamically but slack value does not change during network traversal. In reality, the slack value also changes dynamically because slack value depends on the packet latency, predecessor latency, number of predecessors.

On comparing STC prioritization with slack prioritization, STC prioritize critical applications and slack prioritize individual packets. Hence slack prioritization can provide higher throughput.

IV. CONCLUSION

Packet scheduling policies critically impact the performance and fairness of NoC. The existing packet scheduling policies like age, round robin and STC based having many advantages and disadvantages. And many limitations on packet scheduling policies are still exist. Stall Time Criticality based and Slack based prioritization having better performance and throughput than age and Round Robin. Because different packets can have vastly different importance to their respective application performance, the applications can be classified based on their criticality. This application aware scheduling mechanism can improve the fairness and performance. In the slack based prioritization mechanism prioritize individual packets based on the criticality. Thereby increase the performance and fairness than all other schemes. The limitation of slack based system is it does not dynamically update slack value. If the slack value of the packet update dynamically, it will improve the performance and fairness of the system.

REFERENCES

[1] Sergio Tota, Mario R. Casu, Luca Macchiarulo, “Implementation Analysis of NoC: A MPSoC Trace Driven Approach”, GLSVLSI ’06 Proceedings of the 16th ACM Great Lakes symposium on VLSI, 2006.

[2] [2] Ville Rantala, TeijoLehtonen, JuhaPlosila, “Network on Chip Routing Algorithms”, TUCS Technical Report No 779, August 2006.

[3] D.N. Jayasimha, Bilal Zafar, YatinHoskote, “On-Chip Interconnection Networks: Why They are Different and How to Compare Them”, 2017.

[4] William J. Dally and Brian Towles, “Route Packets, Not Wires: On-Chip Interconnection Networks”, DAC 2001, ACM.

[5] Terry Tao Ye, Luca Benini, Giovanni De Micheli, “Packetization and Routing Analysis of On-Chip Multiprocessor Networks”, Elsevier Science 11 September 2003.

[6] Dennis Abts and Deborah Weisser, “Age-Based Packet Arbitration in Large-Radix k-ary n-cubes”, ACM 2007.

[7] FalkoGuderian, Erik Fischer, Markus Winter, Gerhard Fettweis, “FAIR RATE PACKET ARBITRATION IN NETWORK-ON-CHIP”, IEEE 2011 september

[8] Suyog K. Dahule, Reetesh V. Golhar, Mangesh D. Ramteke, “The Behavior of Round Robin Arbiter in NOC Architecture”, IJEIT, November 2013.

[9] Jae W. Lee, Man Cheuk Ng, KrsteAsanovic, “Globally-Synchronized Frames for Guaranteed Quality-of-Service in On-Chip Networks”, 35th IEEE/ACM International Symposium on Computer Architecture (ISCA),2008.

[10] O. Mutlu and T. Moscibroda. “Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors”. In MICRO-41, 2007.

[11] Reetuparna Das, OnurMutlu, Thomas Moscibroda, Chita R. Das, “Application-Aware Prioritization Mechanisms for On-Chip Networks”, MICRO09, ACM 2009.

[12] Reetuparna Das, OnurMutlu, Thomas Moscibroda, Chita R. Das, “Aergia: Exploiting Packet Latency Slack in On-Chip Networks”, ISCA10, ACM 2010, June 1923.

[13] Asit K. Mishra, OnurMutlu, Chita R. Das, “A Heterogeneous Multiple Network-On-Chip Design: An Application-Aware Approach”, DAC 13, ACM 2013.

[14] Daniel Andreasson and Shashi Kumar, “Slack-Time Aware Routing in NoC Systems”, IEEE International Symposium on Circuits and Systems 2005.

[15] W. J. Dally and B. Towles, "Principles and Practices of Interconnection Networks", Morgan Kaufmann, 2003.