Routing Algorithms, Process Model for Quality of Services (QoS) and Architectures for Two-Dimensional 4×4 Mesh Topology Network-on-Chip

Nauman Jalil, Adnan Qureshi, Furqan Khan, and Sohaib Ayyaz Qazi

***Abstract*—In this paper we have provided routing algorithms, process model for quality of service (QoS) and architecture for new Timer based adaptive routing algorithm for a generic network, based on a two-dimensional mesh topology. Compared to previous work, our proposed work has provided with details of routing algorithms and process model for four class of services used in on-chip networks. The QoS requirements (delay and throughput) for each class of service has met for deterministic XY wormhole routing and further improved for by Timer based adaptive routing algorithm. Simulation results show the improvement achieved by Timer based adaptive routing algorithm as compared to deterministic XY wormhole routing which is based on shortest path.**

***Index Terms*—Networks-on-Chip (NoC), System-on-Chip (SoC), On-Chip Communication, Quality of Service (QoS)**

I. Introduction

With the advancement in the VLSI technologies, where many cores are now being embedded in a single chip, the design complexity of connecting them globally has increased considerably. Therefore, network on chip has been proposed and will be one of the preferred interconnection methods for high performance system. To solve system on chip (SoC) design challenges, networks-on-chip (NoC) has been accepted to solve these design challenges. One of the key researches of Network on chip design is the routing algorithms. For achieving high performance, good routing algorithms are needed; therefore, we have presented Timer based adaptive routing algorithm for a 2-D mesh topology and showed the improvement it has achieved against simple XY deterministic wormhole routing algorithm as in [1]-[2].

In modern system-on-chip architectures comprise a heterogeneous IP core such as central processing unit, DSP processor, Video controller and several embedded memories. Each of these processing elements (PE) is connected to a local router, which connects the node to the neighboring nodes via a NoC [3]. When a node wants to transmit a packet to a destination node, the packet is first generated by a node and then transmitted to the physical link, via a network interface (NI) attached to a router. The packet is then stored in the input queue of the router, before the router start servicing it. The router serve the packet when it turns come, and makes a routing decision based on the information on it packet header, it then allocates a channel and traverse the switch fabric. After the packet is served, the packet moves to the next router and the process continues until the packet is delivered to its final destination.

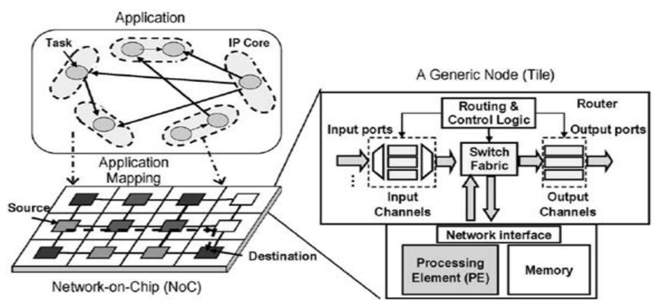

Fig. 1. System on chip architecture.

II. Literature Review

Since the advent of multi-core chips more than a decade ago, a lot of research work has been done in the area of network-on-chip. Latency and throughput of the network is highly reliant on the topology. It determines the number of hops a packet must traverse. It is shown that cluster mesh (CMesh) and MinRoot topologies has achieved better network latency and energy consumption, while obtaining slight area overhead and does not increase the complexity of architectures [4].

In MESH architecture, each IP element is attached to a single router node. This means number of router node is equal to number of IP nodes $R=N$, where R is the number of routers and N is the number of IP nodes. The total number of bi-directional links is equal to 3n2 – 2n and the diameter is equal to 2n [5].

In SPIN (scalable, programmable, integrated network) each node has four children. The size of the network can be evaluated as $N\log(N)/8$. In this architecture the total number of routers can be evaluated by $R=3N/4$, where N is the total number of IP functional nodes [6].

The BFT architecture is very similar to SPIN architecture. They both belong to the same fat-tree architectures and have similar concepts. The IP units are situated in the leaves of the fat-tree architecture and the routers in the nodes of the tree [1].

The number of router converges by using the following equation:

$$R = N/2^* [(N/4+1/2^*N/4+1/4^*N/4+\ldots)] \qquad [1]$$

In CMesh topology allows multiple IP's to be connected

Manuscript received January 26, 2011; revised May 12, 2011.
The authors are with the Department of Electrical Engineering, CASE University, Islamabad, Pakistan (e-mail: nauman_jaleel@yahoo.com).

with a single router. Now, each router consumes more area and requires large number of buffers, 20 input buffers for every 4 IP's connected with a router. Whereas, only 8 input buffers are required for Mesh topology [4].

The MinRoot topology is similar to CMesh, only an additional router is added in the middle, so to distribute the load among the four routers. Now, the traffic among the 8 nodes above does not need to flow among the 8 nodes below.

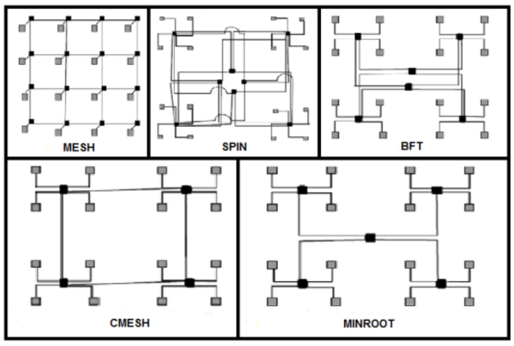

Fig. 2. NoC topologies.

With optimized topology and route allocation certainly saves up the buffers allocation occupied by the interconnecting nodes and provides less congestion on each link of the interconnecting nodes [7].

While numerous routing algorithms have been proposed, the most commonly used routing algorithm in on-chip networks is Dimension Ordered Routing (DOR) due to its simplicity. Dimension ordered routing is an example of a deterministic routing algorithm, in which all messages from node A to B will always traverse the same path. With DOR, a message traverses the network dimension by dimension, reaching the ordinate matching its destination before switching to the next dimension. In a 2-dimensional topology such as the mesh *X*-*Y* dimension-ordered routing sends packets along the *X*-dimension first, followed by the *Y*-dimension. A packet travelling from (0, 0) to (2, 3) will first traverse 2 hops along the *X*-dimension, arriving at (2, 0), before traversing 3 hops along the *Y*-dimension to its destination.

A more sophisticated routing algorithm can be adaptive, in which the path a message takes from A to B depends on network traffic situation. For instance, a message can be initially following the *X*-*Y* route and see congestion at (1, 0)'s east outgoing link. Due to this congestion, the message will instead choose to take the north outgoing link towards the destination [8].

Wormhole flow control allows flits to be transmitted by the current node as soon as there is sufficient buffering space is available at the input queue of the router wormhole flow control allocates storage and bandwidth on the basis of flit size rather than the entire packet [9].

In the example below, the size of the buffers at each router is 2 flits. When the header flit experiences contention while traveling form node 1 to node 2, the remaining flits are stalled at node 0, as there is no buffer space for the flit are available at node 1. But the channel is still occupied by the packet as shown in grey. By using wormhole flow control technique, it allows a flit to leave a router as soon as buffer space is available at the router. Because of high area and power constraints of on-chip networks, wormhole flow control is the most predominated technique used so far.

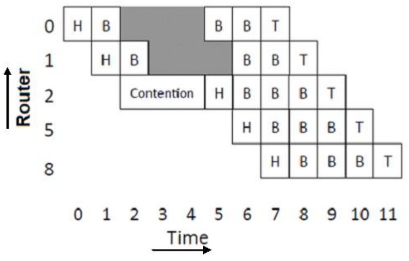

Fig. 3. Wormhole example.

The separate queues in the router are actually the virtual channels (VC) of the router. Multiple virtual channels of the routers input port shares the same physical link between two routers. By allocating multiple separate queues for each input port of the router, the packets head-of-line blocking is reduced. After each cycle the virtual channels of each input port arbitrate for the bandwidth of the physical link. When a packet occupying a virtual channel and it is blocked, other packets can still travel across the physical link through other virtual channels of the routers input port. Therefore, with the help of virtual channels we can increase the network throughput and extend the utilization of the physical links.

Virtual channels were first proposed by Dally with wormhole flow control, but it can be applied to all other flow control techniques, to decrease the head-of lineblocking. For example, we can use circuit switching on virtual channels instead on physical channel, so when the message traverses through the links, it reserves a series of virtual channels instead of physical links. The virtual channels uses time-multiplexing on the cycle-by-cycle basis for transmission onto the physical link, also called virtual circuit switching. Similarly, we can use virtual channels for store-and-forward flow control, by assigning one virtual channel per each packet buffer queues. Virtual channels are time multiplexed onto the physical link packet-by-packet basis. As network-on-chip designs widely adopt wormhole flow control technique for its small area, therefore when we mention virtual channel flow control, it is assumed that it is applied to wormhole flow control [10].

III. Routing Algorithms, Process Model and Architectures for 2-D Mesh NoC

We proposed a Timer based adaptive routing algorithm for a 2-D mesh topology and shown that how our routing algorithm has achieve improved simulation results. Four types of service levels are used and also provide Quality of Services for each service level. First, simulation results are taken by applying deterministic XY routing by using wormhole flow control technique [9] and then by adaptive routing algorithm and showed its improvement against deterministic XY routing.

A. *Mesh Topology*

For our analysis we have used simple 2-D mesh topology shown in Fig 4. Every node is attached to only one router. Each router has five outgoing ports, four for routing the packets in four different directions (North, South, West and East) and one port for the node attached to it.

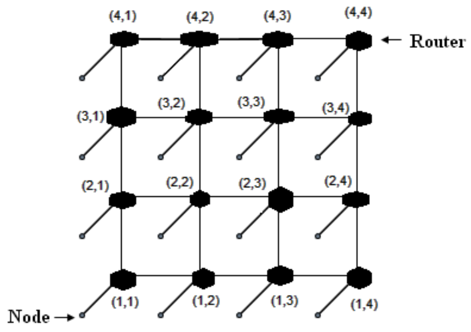

Fig. 4. Mesh topology

Each router is distinguished by the *X* coordinate and *Y* coordinates. The *X* coordinate for the routers in the bottom row are assigned 1. The *X* coordinate for the row above it are 2, 3 and 4. Similarly, the *Y* coordinate for the routers in the left most columns are assigned 1. The *Y* coordinate for the columns to its right are 2, 3 and 4.

B. Node Model

Each node generates four types of traffic; signaling, real time, read block/write block and block transfer.

1. Packet format

The first four bits are for the destination node for *X* and *Y* coordinates, 2 bits each. Similarly, the next four bits are for the source node. The SL bits are for the four types of service level as defined below:

Signaling	00
Read Block/ Write Block	01
Real Time	10
Block Transfer	11

Dest_Xaxis (2bits)	Dest_axis (2bits)	Src_Xaxis (2bits)	Src_Yaxis (2bits)	SL (2bits)

Fig. 5. Packet format.

2. Node architecture

There are four processor modules for each type of service level as shown in the figure, which generates the traffic for each service level and transmits them to the appropriate destination. It also receives the packets for each service level from the router's input ports and demultiplexes them to the appropriate service level.

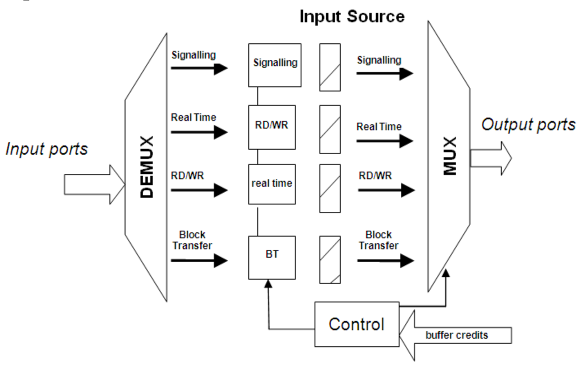

Fig. 6. Node architecture.

The buffer credit receives the signaling information about the current state of the input buffers for each service level at the router input. Whenever, a flit is transmitted from the router's input buffer and the buffer space is freed from the router, a buffer credit signal is transmitted to the node, so that it can transmit further flits to the router.

3. Node process model

The node receives packets from the four sources and from the input port of the router which receives packets from the router for each of the four services. There are separate signaling wires which receives buffer credits signaling information. Whenever, a node receives a packet from either of these sources an interrupt is generated and the node serves the required request.

4. Algorithm of the node process model

The node process model receives nine different types of interrupt. But at any particular instant of time, the node only process one interrupt. The first four interrupts in the algorithm are from the four service levels of the node. The node transmits each packet for each service level, flit-by-flit to the router. Before sending the flit to the router, the node first checks if there is buffer space available at the input buffer of the router. For each service level there are separate input buffers, therefore there are separate counters for each service level. After each flit is transmitted buffer counter is decremented. If the buffers counter reaches the value of zero, all further flit are blocked by the node and wait for the buffer credit signal from the router. For every first flit of the packet the node also attaches the header of the packet, which includes addresses of source and destination and the type of service level.

The next four interrupt in the algorithm are from the router's output port, which delivers these packets from the intended sources to the required destination. The node concatenates all the flits received by the router and delivers them to the required service level. The node also measures the latency for each packet and the total number of packets received and save them in file. The last interrupt in the algorithm is the buffer credit signal. The router sends these signals to the node whenever a buffer space is available at the router, for each service level.

C. Router Model

In Fig. 6 each router has five input and output ports and each port has four virtual channels for each four types of service level [1]. By using these virtual channels the traffic of one service will not block the traffic of another service.

1. Router architecture

The *CRT* buffer maintains the routing information of the packets to be transmitted to the destination. As the packets are divided into flits, so routing information need to be safe in a buffer for subsequent flits to be followed for each packet.

The NBS buffer maintains the current buffer state of the next input port of the router, so that the source does not overflow the buffers of the input ports. At each input port, there are limited amount of buffers available. Therefore, before transmitting a flit, the router first check the buffer space at the next input port. Where ever the buffer space is available at the routers input port, the buffer credit signal is transmitted by the Control Routing Block of the router, which is then received by the control block at the routers output port.

The *CSIP* table maintains the present state of round robin scheduling used for each service level at each output port.

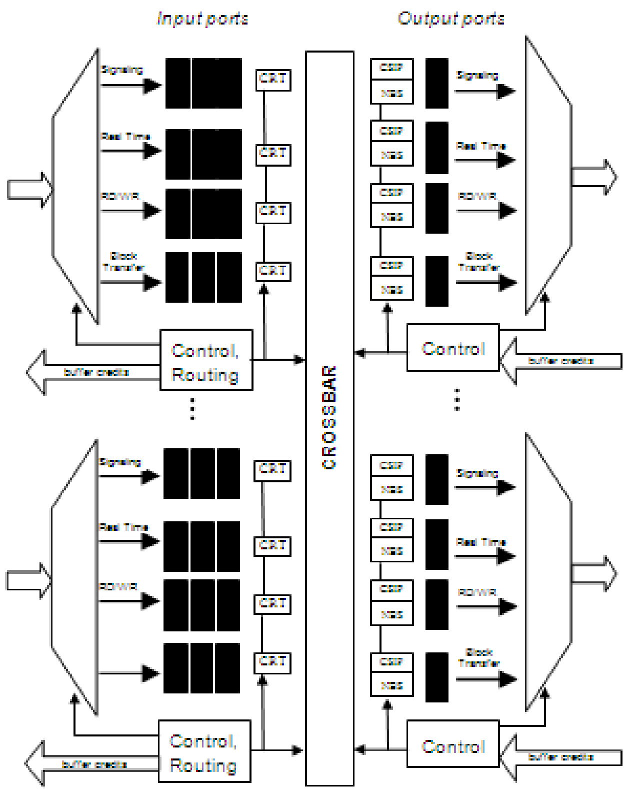

Fig. 7. Router architecture.

2. Router process model

The process model begins form the init module, from here the variables are first initialized and the router process model starts. In the start the router process model is blocked in the idle state, when a signaling flit arrives and if the link is available, a transition signaling up occurs to signaling state. As long signaling link is not blocked, router serves only the signaling flits and whenever the link is blocked due to insufficient space available at the next router input, transition link not free is occurred to queue packet state, all further flits are queued and the transition signaling down is occurred to the idle state.

Similarly, if real time flit arrives and router does not have signaling flits to serve, the transition real time up occurs to real time state and serve all it flits. If signaling flits arrive during real time state, the transition again goes back to signaling state and first serves all it flits and then come back to real time state. In this way the router can preserve the signaling priority. As long real time link or signaling link is not blocked, router serves only the real time and signaling flits respectively, whenever either of the link is blocked due to insufficient space available at the next router input, transition link not free is occurred to queue packet state, all further flits are queued and the transition real time down or signaling down is occurred and then back to the idle state.

Similarly, if read block/ write block flit arrives and router does not have signaling and real time flits to serve, the transition read block/ write block up occurs to RD_WR state and serve all it flits. If signaling flits arrive during RD_WR state, the transition again goes back to signaling state and first serves all it flits and then come back to RD_WR state. In this way the router can preserve the signaling priority. Similarly, if real time flits arrive during RD_WR state, the transition again goes back to real time state and first serves all it flits and then come back to RD_WR state. In this way the router can preserve the real time priority. As long signaling link, real time link and RD_WR link is not blocked, router serves only the signaling flits, real time flits and RD_WR flits respectively, whenever the link is blocked due to insufficient space available at the next router input, transition link not free is occurred to queue packet state, all further flits are queued and the transition signaling down, real time down or RD_WR down is occurred and then back to the idle state.

Similarly, if block transfer flit arrives and router does not have signaling, real time and read block/ write block flits to serve, the transition block transfer up occurs to block transfer state and serve all it flits. If signaling flits arrive during block transfer state, the transition again goes back to signaling state and first serves all it flits and then come back to block transfer state. In this way the router can preserve the signaling priority. Similarly, if real time flits arrive during block transfer state, the transition again goes back to real time state and first serves all it flits and then come back to block transfer state. In this way the router can preserve the real time priority. As long signaling link, real time link, RD_WR link and block transfer is not blocked, router serves only the signaling flits, real time flits, RD_WR flits and block transfer respectively, whenever the link is blocked due to insufficient space available at the next router input, transition link not free is occurred to queue packet state, all further flits are queued and the transition signaling down, real time down, RD_WR down or BT down is occurred and then back to the idle state.

3. Router process model for adaptive routing algorithm

In the init module the variables are first initialized and the router process model starts. The router process model starts from the idle state, when a signaling flit arrives and if link is available, a transition signaling up occurs to signaling state. As long signaling link is not blocked, router serves only the signaling flits and whenever the link is blocked, transition to Link not free is occurred, all the further flits are queued and the timer interrupt is set and then the transition to signaling down is occurred and then back to the idle state.

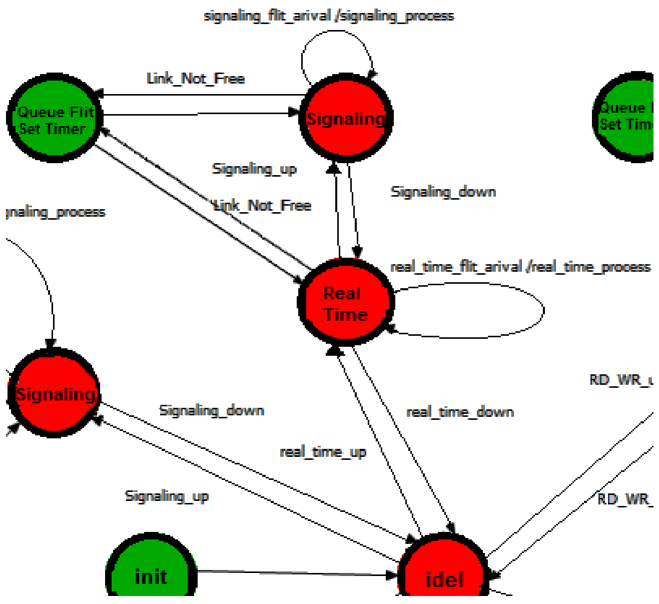

Fig. 8. Router process model (part1).

Similarly, if real time flit arrives and router does not have signaling flits to serve, the transition real time up occurs to real time state and serve all it flits. If signaling flits arrive during real time state, the transition again goes back to signaling state and first serves all it flits and then come back to real time state. In this way the router can preserve the signaling priority. As long real time or signaling link is not blocked, router serves only the real time and signaling flits respectively and whenever either of the link is blocked, transition to Link not free is occurred, all the further flits are queued and the timer interrupt is set and then the transition to real time down or signaling down is occurred and then back to the idle state as shown in Fig. 8.

Similarly, if read block/write block flit arrives and router does not have signaling and real time flits to serve, the transition read block/write block up occurs to RD_WR state and serve all it flits. If signaling flits arrive during RD_WR state, the transition again goes back to signaling state and first serves all it flits and then come back to RD_WR state. In this way the router can preserve the signaling priority. Similarly, if real time flits arrive during RD_WR state, the transition again goes back to real time state and first serves all it flits and then come back to RD_WR state. In this way the router can preserve the real time priority. As long RD_WR, real time or signaling link is not blocked, router serves only the RD_WR, real time and signaling flits respectively and whenever any of the link is blocked, transition to Link not free is occurred, all the further flits are queued and the timer interrupt is set and then the transition to the idle state is occurred as shown in Fig. 9.

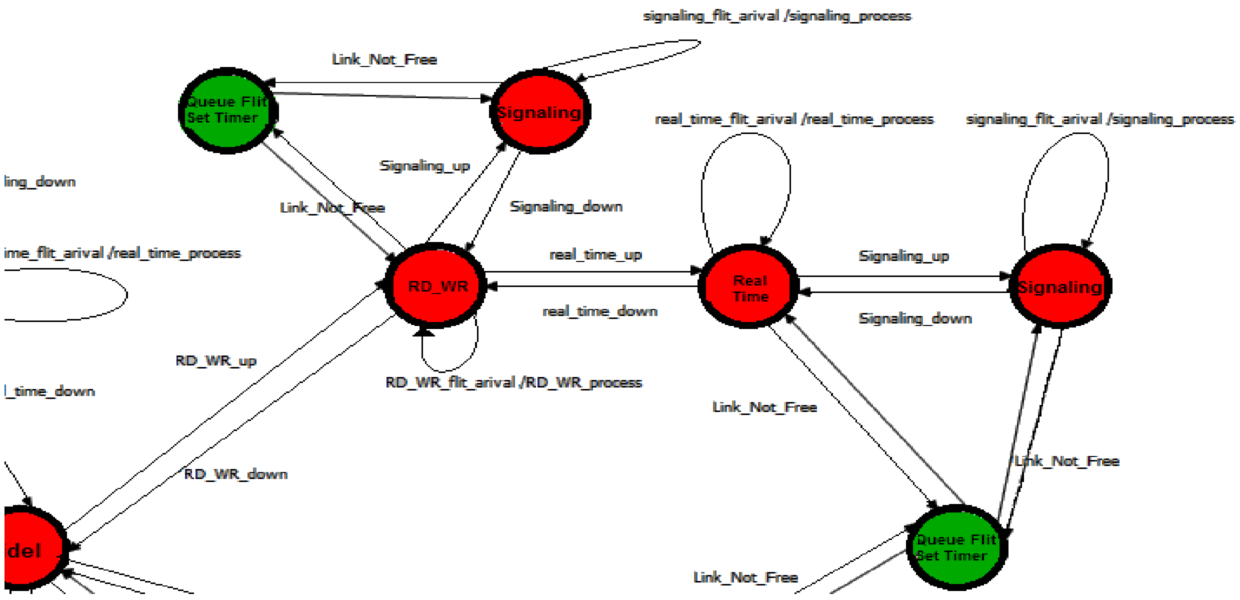

Fig. 9. Router process model (part2).

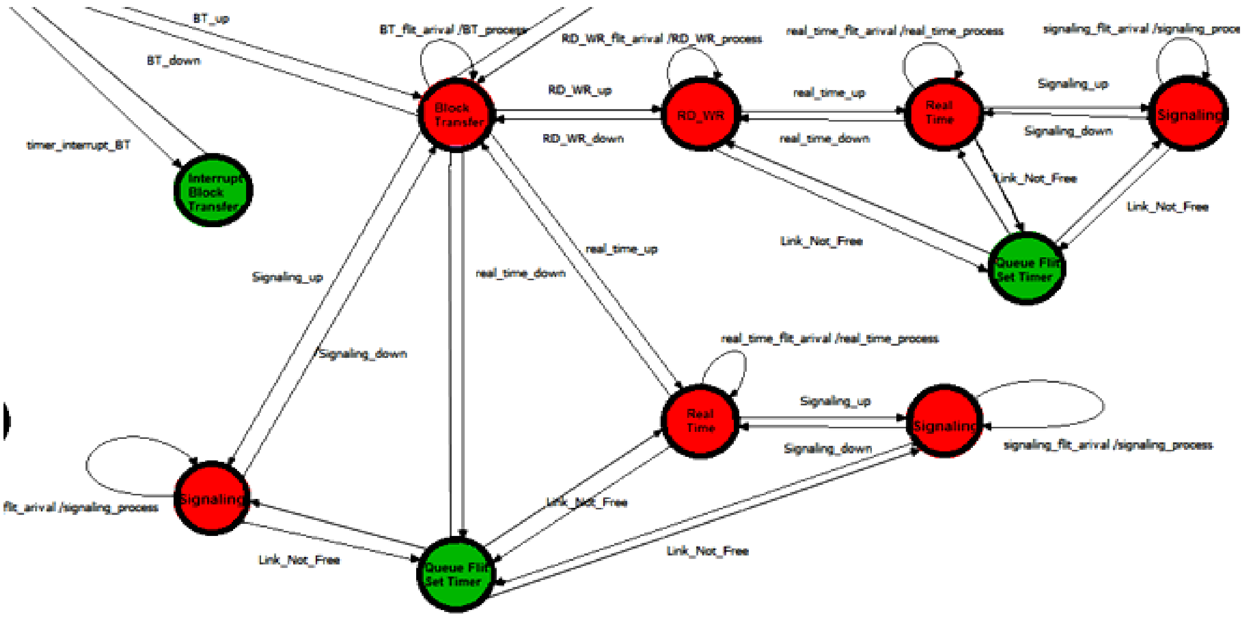

Fig. 10. Router process model (part3).

Similarly, if block transfer flit arrives and router does not have signaling, real time and read block/ write block flits to serve, the transition block transfer up occurs to block transfer state and serve all it flits. If signaling flits arrive during block transfer state, the transition again goes back to signaling state and first serves all it flits and then come back to RD_WR state. In this way the router can preserve the signaling priority. Similarly, if real time flits arrive during block transfer state, the transition again goes back to real time state and first serves all it flits and then come back to block transfer state. In this way the router can preserve the real time priority. As long Block Transfer, RD_WR, real time or signaling link is not blocked, router serves only the Block Transfer, RD_WR, real time and signaling flits respectively and whenever any of the link is blocked, transition to Link not free is occurred, all the further flits are queued and the timer interrupt is set and then the transition to the idle state is occurred as shown in Fig 10.

Whenever, the timer is expired for any of the service level, signaling, real time, RD_WB and block transfer, the required interrupt is generated, the signaling interrupt, real time interrupt, RD_WB interrupt and block transfer interrupt respectively. The required interrupt state checks whichever the link is available and select the appropriate link which available.

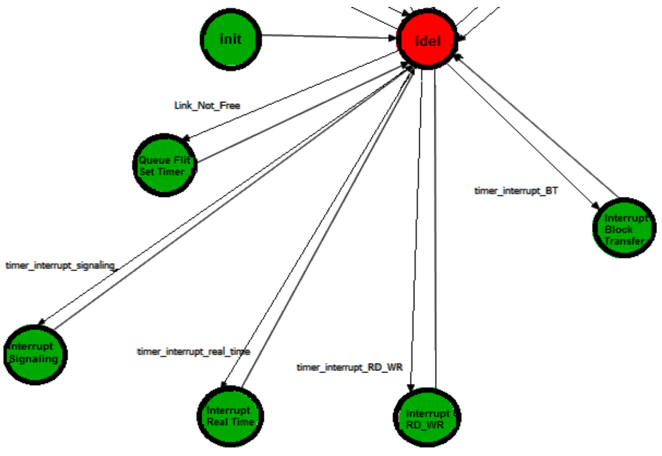

Fig. 11. Router process model (part4).

4. Adaptive routing algorithm

When the flit arrives, it first checks whether the link is available or not. If link is available the flit is routed in the appropriate direction by using simple XY routing. If link is not available the flit is queued and the threshold timer is set. If the link is free before the timer is expired, then the flit is routed in the XY direction. Otherwise, if link is not free before the timer is expired, then alternate link is selected which is available. With the help of this adaptive routing algorithm, the flits does not need to be buffered in the queue for long duration, if the timer is expired then alternate link can be selected, if available.

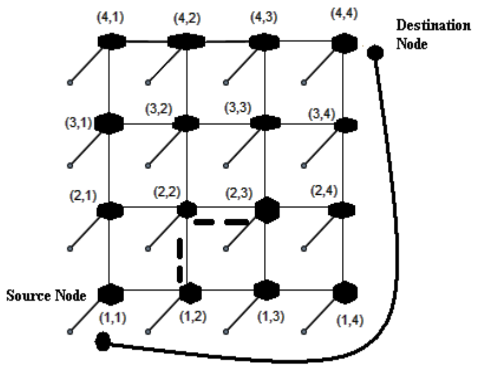

Fig. 12. Mesh topology architecture.

As, the bottom link (from node (1,1) to node (1,4)) and the link at the right most(from node (1,4) to node (4,4)) is all occupied by source node (1,1) and destination node (4,4), therefore the packets needed to be transmitted by source node (1,2) to destination node (2,3) is blocked, because the link between node (1,2) and node (1,3) is occupied, whereas the north link (form node (1,2) to node (2,2)) is available. Therefore, with our new adaptive algorithm, the alternate link (form node (1, 2) to node (2, 2) can be selected (as shown by dotted lines), if the link in not freed before the timer expires and thus increases the throughput of the network, considerably.

The value of the timer interrupt is an important parameter and its value should be evaluated properly for each service level. In Fig. 13, if the value of the timer interrupt is not properly assigned the packets may take longer routes, in such case, more links are reserved and it may decrease the throughput of the network.

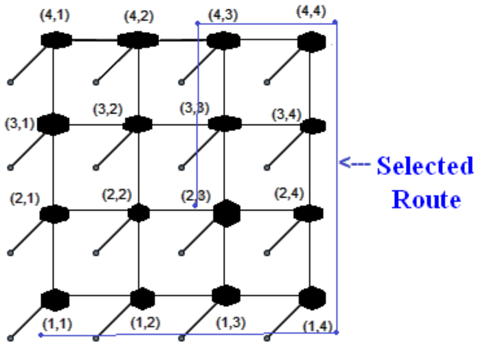

Fig. 13. Mesh topology architecture

I. Simulation Results

We have provided the latency and throughput for each service level. The latency of the packet is measured as follows:

Latency = (Time when packet is created) – (Time when packet is received at the destination).

The latency of a packet includes the following parameters:

- Transmission Time
- Propagation Delay
- Queue processing Delay
- Router Processing Time

Throughput = Number of packets received per second

TABLE I: Summarized Results and Comparison

Routing	Signalling		Read Block/Write Block	
Algorithm	Latency(ns)	Throughput	Latency(ns)	Throughput
XY	5.8	5301	6.1	16002
Adaptive	5.1	6507	5.71	16770
Routing	Real Time		Block Transfer	
Algorithm	Latency(ns)	Throughput	Latency(ns)	Throughput
XY	45	1546	8.1	81
Adaptive	38	2050	5.96	144

Following are the simulation results that we have obtained by using our adaptive routing algorithm by varying the threshold for each type of traffic. The threshold is the value of the timer interrupt. Form the plots we can see that for different threshold values, the latency and throughput varies for adaptive routing algorithm. For XY routing it is constant, because their routes are predefined and does not vary with threshold. Therefore, an optimal threshold needs to be selected, in order to achieve best results, which are shown in Table I.

a) Simulation results for signaling

In Fig. 14 maximum throughput and minimum latency we have achieved for signaling traffic is 6,507 and 5.1 ns respectively, by using our new adaptive routing algorithm. But when the threshold value is very small the throughput has decreased to 4,699 and latency has increased to 9ns. Therefore, the value of threshold should be adjusted properly.

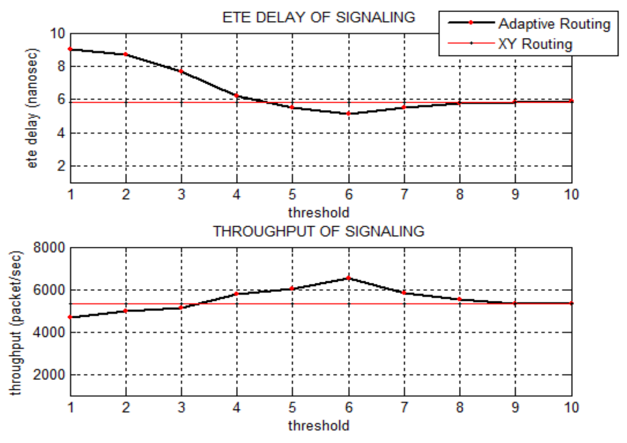

Fig. 14. Simulation result for signaling.

b) Simulation results for read block/ write block

In Fig. 15 maximum throughput and minimum latency we have achieved for read block/ write block is 16,770 and 6.3 ns respectively, by using our new adaptive routing algorithm. But when the threshold value is very small the throughput has decreased to 14,001 and latency has increased to 8.9 ns.

c) Simulation results for real time

In Fig. 16 maximum throughput and minimum latency we have achieved for real time traffic is 2,050 µs and 38 µs respectively, by using our new adaptive routing algorithm. But when the threshold value is very small the throughput has decreased to 1,555 and latency has increased to 55 µs.

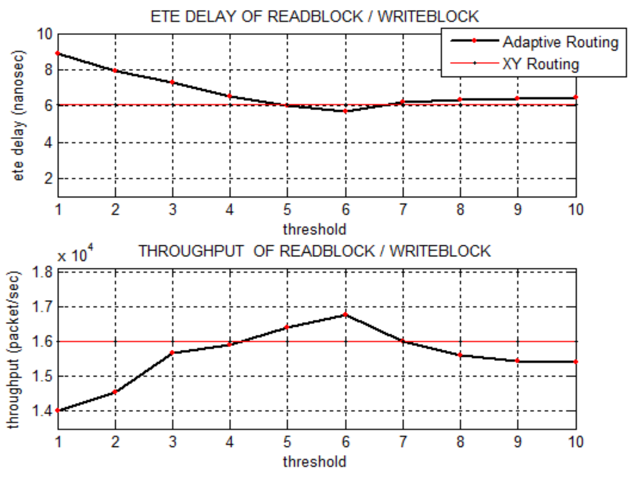

Fig. 15. Simulation result for RD/WB.

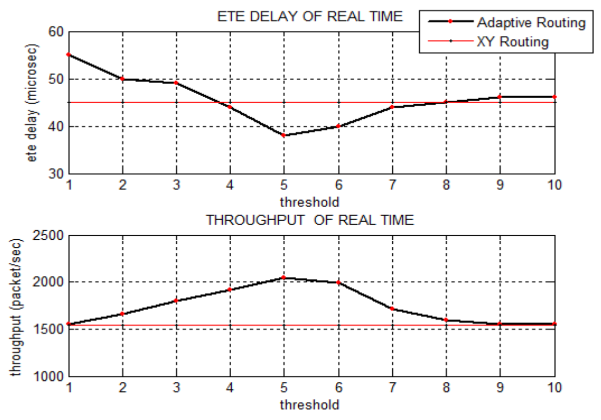

Fig. 16. Simulation result for real time.

d) Simulation results for block transfer

In Fig. 17 maximum throughput and minimum latency we have achieved for block transfer is 144 ms and 5.96 ms respectively, by using our new adaptive routing algorithm.

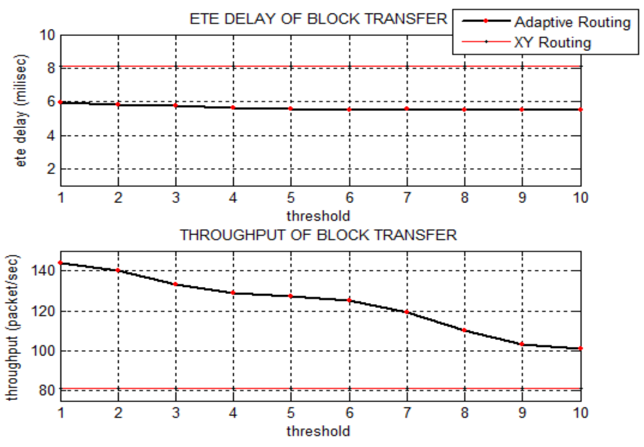

Fig. 17. Simulation result for block transfer.

II. Conclusion and Future Work

In our research work, we have proposed our new adaptive routing algorithm and have shown its superiority over simple deterministic XY wormhole routing. We have provided the architecture and process model for both router and node, used in the 4x4 mesh topology NoC. We have also provided with in-depth details for routing algorithm, for both deterministic XY wormhole routing and adaptive routing algorithm. Since, NoC is relatively a new field and very little research work has been done in adaptive routing. Therefore, over proposed work has certainly provided with the details for routing algorithm for network on chip.

Further research can be done in this area, by implementing the architecture of the network in FPGA or ASICS and see what area and frequency we can further improve to increase the throughput and minimize the latency of the network.

References

[1] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "QNoC: QoS architecture and design process for network on chip," *Journal of Systems Architecture*, vol. 50, no. 2-3, 2003.

[2] W. J. Dally and B. Towles, "Route packets, not wires: onchip interconnection network," in *Proc. of Design Automation Conference*, Las Vegas, 2001, pp. 684-689.

[3] R. Marculescu, U. Y. Ogras, and L.-S. Peh, "Natalie Enright Jerger, and Yatin Hoskote, Outstanding Research Problems in NoC Design," *IEEE Transactions on computer-aided design of integrated circuits and systems*, vol. 28, no. 1, Jan 2009

[4] M. A. J. Jamali and A. Khademzadeh, "MinRoot and CMesh: Interconnection Architectures for Network-on-Chip Systems," *World Academy of Science, Engineering and Technology*, vol.54, 2009.

[5] S. Kumar *et al.*, "A Network on Chip Architecture and Design Methodology," in *Proc. Int'l Symp. VLSI (ISVLSI)*, pp. 117-124, 2002.

[6] J. Hennessey and D. Patterson, *Computer Architecture: A Quantitative Approach. Morgan Kaufmann*, Ch 8, pp. 822-830, 2003.

[7] M. Jun, S. Yoo, and E.-Y. Chung, "Mixed Integer Linear Programming-based Optimal Topology Synthesis of Cascaded Crossbar Switches*," IEEE Transactions*, vol. 7B, no. 1, 2008.

[8] W. Zhang, L. Hou, J. Wang, S. Geng, and W. Wu, "Comparison Research between XY and Odd-Even Routing Algorithm of a 2-Dimension 3X3 Mesh Topology Network-on-Chip," *IEEE Computer Society, Global Congress on Intelligent Systems,* 2009.

[9] W. J. Dally, "Virtual-channel flow control," In *Proceedings of the International Symposium on Computer Architecture*, DOI: 10.1109/71.127260, 1990.

[10] P. P. Pande, C. Grecu, A. Ivanov, and R. Saleh, "Design of a Switch for Network on Chip Applications," *Proc. Int'l Symp. Circuits and Systems (ISCAS)*, vol. 5, pp. 217-220, May 2003.