An Adaptive Routing Algorithm for on-chip 2D Mesh Network with an Efficient Buffer Allocation Scheme

Syeda Tanjila Atik[1] , M.M. Imran[2] , Julkar N Mahi[3] , Jenia A Jeba[4] , Z I Chowdhury[5] , M S Kaiser[6]
Institute of Information Technology, Jahangirnagar University, Savar, Dhaka-1342, Bangladesh
Dept. of Computer Science & Engineering, Daffodil International University, Dhaka-1205, Bangladesh
Email: { tanjila.tanji15, polboy777, mahi.1992, zeniazeba }@gmail.com, { zic, mskaiser }@juniv.edu

Abstract— **Network-on-Chip (NoC) is thought to be an effective packet-switched on chip arrangement for System-on-Chip (SoC) paradigm. Using router in a NoC, higher throughput is facilitated which is desirable for dealing with the difficulty of current systems. For proper usage of the communication bandwidth and to diminish transportation delay, an intelligent routing mechanism is required. Also a proper buffer management is essential to reduce packet drops and area overhead. Several routing algorithms and buffering techniques are proposed so far. In this paper, we propose a modified XY routing algorithm coupled with an on-demand buffer allocation concept for a mesh on-chip network and compare the performance of our algorithm with other frequently used algorithm named as Odd-Even and DyAD routing. Simulation results indicate that the proposed algorithm achieves better performance than traditional XY-routing and other algorithms in terms of latency and throughput. Again, the effect of using different number of virtual channels in the router buffer is also studied here.**

Keywords— 2D mesh topology, Network on Chip (NoC), Odd Even (OE), NIRGAM simulator.

I. INTRODUCTION

Network on chip (NoC) refers to a communication subsystem on an integrated circuit which can achieve a high level of parallelism because all links in the NoC can be operated concurrently on different data packets. Expanding integration prompts a circumstance where commonly used bus structure ends up noticeably blocked and expanded capacitance which posture physical issues. In NoC architecture, the main network components are wires and routers. Processors, memories and other IP-blocks (Intellectual Property) are linked to routers [1]. Routers act as switches that receive records at one of the inputs and outputs most effective one by one [2].

Routing algorithm is an important design concept of NoC which is responsible for determining an efficient route for the data or packets to be transferred from source to destination [3]. For area constrained NoC design, if the input buffer size can be abridged then the number of Virtual channels (VCs) will also be reduced which plays a key role in defining the overall performance of the network.

NoC architecture can be adjusted for a particular application to accomplish best vitality, execution and cost trade-offs [4]. The design has to meet some requirements such as reduced latency, guaranteed throughput, sufficient transfer capacity etc. More delays will be produced during packet transmission when the network is overcrowding. In this situation, by introducing an effective routing to make a harmony between the time postponement and throughput rate turns into the key issue [5]. Furthermore, proper bandwidth management has become a key problem because of the increasing number of internet users [6]. Also, the buffer space in the router should be carefully allocated as it is directly proportional to area overhead.

In this paper we propose and execute the application of our modified XY routing algorithm in 2D mesh topology in addition with an efficient priority based buffer assignment scheme to achieve reduced latency and desired throughput in a wired on-chip network. We also observe the effect of using different number of virtual channels in the buffer block in terms of latency and packet drops.

The rest of the part of this paper is stated as follows: In section II, a summary of the related researches are given, in section III, the system model is described, the proposed routing algorithm is explained section IV, simulation results are shown in section V, section VII finally states the conclusion.

II. RELATED WORKS

Maximum of the NoC designs proposed earlier depend totally on a static community configuration using constant routing algorithms and buffer placements Those are not adequate in managing hard-to-foresee framework requirement, for example as a result of changing client needs or differing workloads.

In [7], authors have proposed a deterministic routing called XY routing where a fixed routing path is always used all over the process following the concept of minimal turning routing. According to this algorithm the data packets will navigate in X-direction towards the destination column. After the destination column is found, the data packets will navigate to the destination node in Y-direction. Thus it is not possible for the packets to move first in Y-direction then in X-direction. So, the routing process is restricted in terms of routing or turning. However, XY routing overloads the center of the network because higher load is experienced by the routers situated nearer to the center of the 2D mesh based network.

Jonathan W. Brown et al. [8] proposed a routing algorithm named Weighted Non-Minimal OddEven (WeNMOE), where information is gathered on the status of the network from adjoining routers and a routing cost is estimated and routes having the lowest estimated cost are selected as final path. Since this algorithm has more choices for possible directions, more hardware would be required to implement WeNMOE on a NoC.

A dynamic virtual channel regulator for NoC is suggested in [9]. Here an incorporated buffer block is used in each port to VCs and therefore, VCs may be designed in terms of quantity and depth during runtime. Here multiple ports can't share the VCs. So, if there remains a faulty port, other existing ports may not reuse the buffers.

III. SYSTEM MODEL

A generic model of an on-chip network is shown in Fig-1

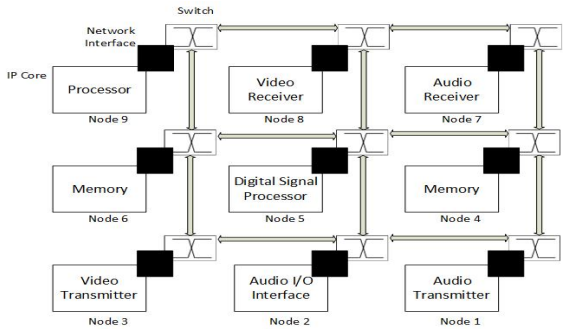

Fig-1: System model of an on-chip network

Here an array of 9 communicating nodes are arranged in a 3×3 mesh network. The nodes can be any IP core i.e. a processor, a video transmitter etc. The switches are embedded in the routers. Assume that, Node 1 with co-ordinate (1,0) acts as a source wants to send a packet to destination node 3 with co-ordinate (3,2). The algorithm we proposed here decides to what direction packets are headed during each phase of the routing. Also how the buffers are allocated to the traffic packets so that the overall performance of the network is increased.

NoC Router Architecture

In our router, there is 5 input and 5 output channels. Virtual Channel Allocator (VCA) allocates the virtual channels at the input buffer to the incoming traffic. Wormhole switching is used here which decides when the packet moves forward from a router. The router architecture for our on-chip network is given in Fig- 2

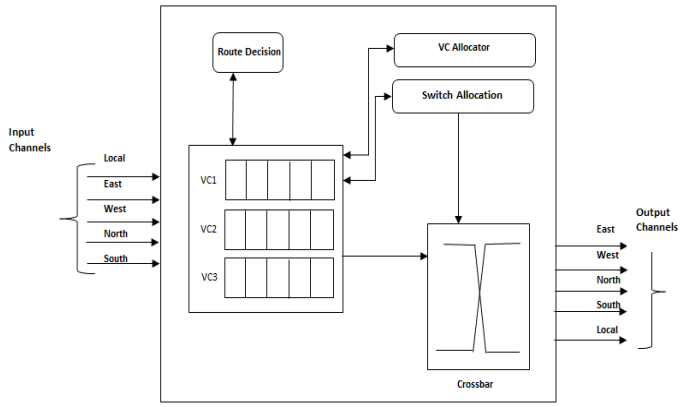

Fig- 2: Model of the Router Architecture

The routing algorithm calculates the routes by making choices locally at every switch depending on the obtainable bandwidth in each direction to the adjacent routers. For transferring the packets, where bandwidth stability is required, a connection is established from the start point of the transaction to the end point. Upon finding the routes, a conforming buffer block is supplied by the routers. When a route is chosen, suitable blocks of buffer need to be arranged depending on the demand of that route. The routes can not be selected at design time in this algorithm, which is in contrast with the XY-routing.

Let's consider a set of directions that packets can be routed, I = {N, E, S, W}, Each k ϵ I, has a weight W_k and obtainable bandwidth BW_k with $BW_k \leq BW_M$, where BW_M is referred to the maximum bandwidth. Every packet p has an end point referred to as destination with coordinates x_d, y_d and BW_P is the required bandwidth.

By using the following equations weights in each direction are allocated:

$$W_N = \begin{cases} BW_N \times D(Y) + BW_M, & D(Y) < 0 \\ 0, & BW_N < BW_P \\ BW_N, & \text{otherwise} \end{cases} \quad (2)$$

$$W_E = \begin{cases} BW_E \times D(X) + BW_M, & D(X) > 0 \\ 0, & BW_E < BW_P \\ BW_E, & \text{otherwise} \end{cases} \quad (3)$$

$$W_S = \begin{cases} BW_S \times D(Y) + BW_M, & D(Y) > 0 \\ 0, & BW_S < BW_P \\ BW_S, & \text{otherwise} \end{cases} \quad (4)$$

$$W_W = \begin{cases} BW_W \times D(X) + BW_M, & D(X) < 0 \\ 0, & BW_W < BW_P \\ BW_W, & \text{otherwise} \end{cases} \quad (5)$$

If there is not enough bandwidth available, the weights are zero. The route R chosen is then to the direction with the highest weight. That is:

$$R = \begin{cases} P, & x = x_d \text{ and } y = y_d \\ k \in \{N, E, S, W\} & \text{else with } W_k = \max(W_k) \end{cases} \quad (6)$$

The proposed algorithm is restricted to topologies like 2-D mesh. However, we can alter it for other random topology (e.g. torus, butterfly etc), but then we have to devise a distance metric for the new topology.

TABLE I
NOTATION USED IN THE PROPOSED ALGORITHM

Symbol	Description
BW_{dir}	Total bandwidth towards current direction
W_{dir}	Weight allocated to a particular direction
$reqBW_{dir}$	Requested Bandwidth to deliver towards current direction
$avBW_{dir}$	offered bandwidth in the current direction for a router

Modified XY Routing Algorithm:

When a packet with an end-point received, **do**
if *D(X) = 0 and D(Y) = 0*, **then**
Channel:= Local
end if
for all output ports P_{dir} **do**,
W_{dir} *:= 0* // initializes all weights to zero
D(X):= |final x – current x|
D(Y) := |final y – current y|
for all P_{dir} with $avBW_{dir} > reqBW_{dir}$, //when bandwidth available
// for output ports directing at East and West
if P_{dir} is facing to the end-point x, **then**
$W_{dir} := avBW_{dir} \times D(X) + BW_{dir}$
else if P_{dir} is facing opposite to the end-point x,
$W_{dir} := avBW_{dir}$
// for output ports directing at North and South
if P_{dir} is facing to the end-point y, **then**
$avBW_{dir} \times D(Y) + BW_{dir}$
else if P_{dir} is facing opposite to the end-point y,
$W_{dir} := avBW_{dir}$
end for // select route for the port consuming highest weight
route:= P_{dir} with max(W_{dir})
Look-up-table:= route //the route is saved in the Look-up-table
return *route*

Flow chart of Buffer allocation scheme

As buffers in NoC routers are important to improve the overall performance of network [10], it is necessary to minimize the size of the buffers and also optimize their usage via a proper management without disrupting the efficiency [11].

A VC [12] is an extra buffer associated with the physical channel, which can accomplish partition between flows having the same physical channel. The flow of buffer allocation is given in Fig-3

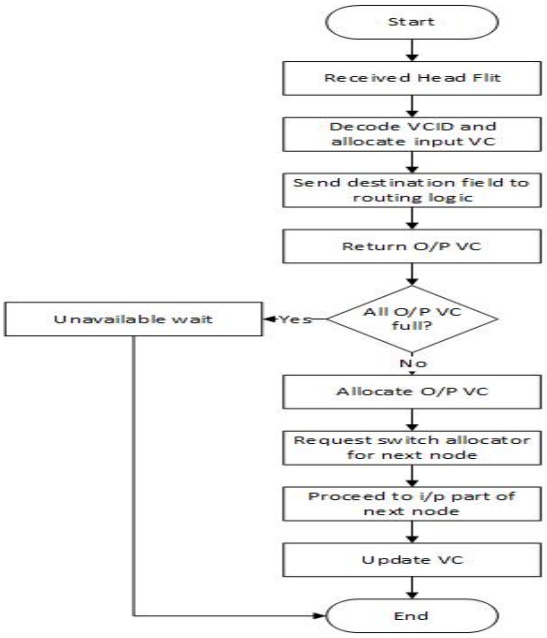

Fig- 3: The flowchart of buffer allocation process

V. SIMULATION RESULTS & ANALYSIS

We have used NIRGAM 2.1 simulator which is extensible and based on modular SystemC. The performance of a NoC design can be analyzed for a user specified application by using this simulator [13]. We simulated our model for 4×4 network where the packet size is varied and length of the packets is exponentially distributed. The buffer size is changed and defined as 3-5 flits per input port. Destinations are uniformly distributed and Poisson traffic is used in running all the simulations.

TABLE II
DESCRIPTION OF THE SIMULATION PARAMETERS

Parameter	Description
Packet size	64 bytes to 256 bytes
No. of clock cycles	2000
Clock frequency	1 GHz
Flit interval between successive flits	2 clock cycles
Flit size (constant)	3 Bytes (1 for head and 2 for data payload)
Traffic Generator	CBR (Constant Bit Rate)
Topology	Mesh 2D 4X4
Source node and Destination node	Node 3 to Node 9
Routing	XY and Modified XY
Per hop latency	1 clock cycle
Virtual channel per port	3 and 6
Traffic workload	Uniform-Random
Warm up period	3000 clock cycle

TABLE III
SIMULATION RESULTS FOR MODIFIED XY ROUTING ALGORITHM FOR 2D MESH OF 4X4 NETWORK

Packet Size (in bytes)	Total Flit Generated	Total Flit Received	Average Latency per flit (in Clock cycles)	Average Throughput (in Gbps)
64	653	457	4.543	12. 272
78	686	433	4.332	12.768
82	725	329	4.765	13.507
96	675	475	4.887	13.669
110	668	428	4.032	13.980
155	751	448	3.298	14.493
176	758	501	3.654	15.277
205	789	493	3.458	15.956
231	850	530	3.465	16.229
256	865	487	3.383	16.875

The simulation result shows that the latency decreased in a significant deviation with increase of the packet size and the throughput is also much better than obtained from XY algorithm which is shown in Fig- 4 and Fig- 5

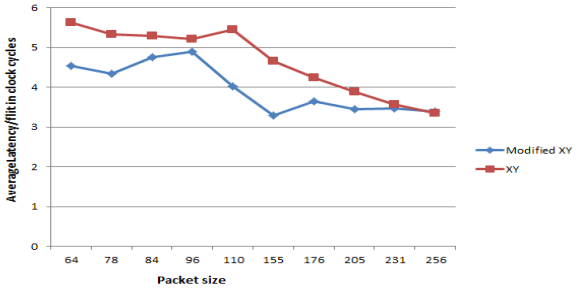

Fig- 4: Comparison of XY and Modified XY in terms of Average Latency Vs Packet size

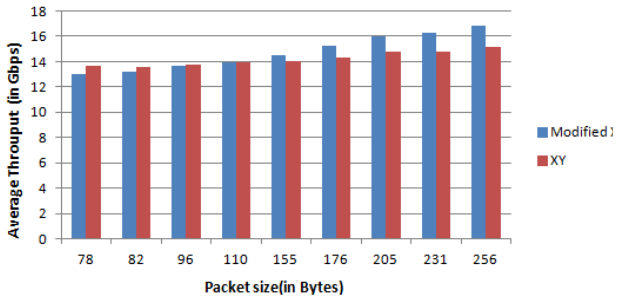

Fig- 5: Comparison in terms of Throughput vs Packet size

Effect of no. of virtual channel buffer on packet drop and average latency per channel

In terms of VCs we found that if the number of VCs in the buffer is less than 5, the packet drop is significantly decreased where the value of latency per packet remains in a considerable state (Fig-6). But for large number of VCs, though the packet drop decreased but the latency increased significantly (Fig-7).

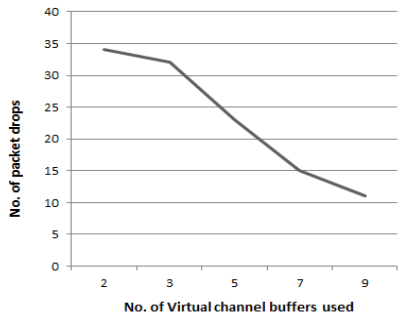

Fig- 6: Effect of using different no. of VCs in Packet drop

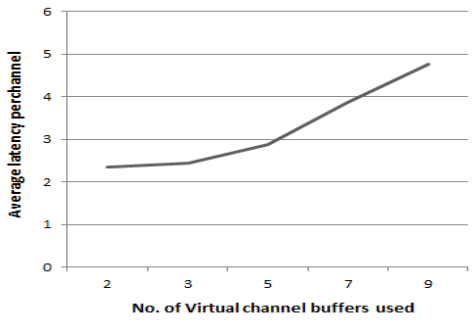

Fig-7: Effect of using different no. of VCs in Average Latency/packet

Link bandwidth utilization by the communicating nodes

As our algorithm chooses the path where there is available bandwidth for the transferring packets, we found the link bandwidth utilization to be in a balanced state. Fig-8 shows the link bandwidth utilization of each of the 16 nodes of our 4×4 network.

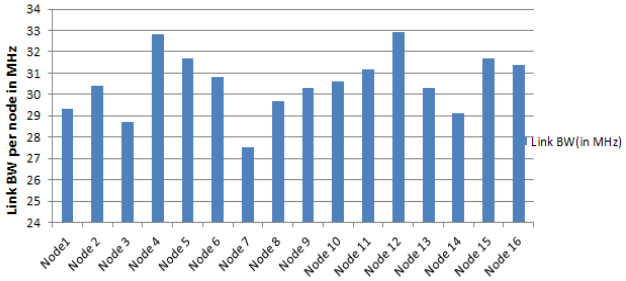

Fig- 8: Utilization of link BW by all nodes in a 4×4 network

Comparison of performance of proposed algorithm with other existing algorithms

Here a performance metric is defined as the ratio between average throughput and average latency of a routing algorithm. We compare the performance of our algorithm with Odd-even and DyAD routing algorithm for some fixed load as 25%, 45% and 75%. Here we measured the load distribution as the total number of flits passed through the nodes. Simulation results show that Modified XY outperforms the OE algorithm in a significant deviation and performs slightly better than DyAD routing for medium load (Fig-9).

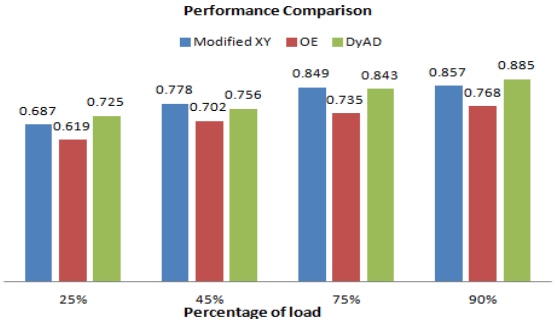

Fig-9: Comparison of OE, DyAD and Modified XY algorithm in terms of values of performance matric

From the simulation results, we found that the modified XY algorithm is capable of reducing most of the limitations of traditional XY routing. In terms of average throughput, we can see Modified XY performs better than XY routing algorithm. However the algorithm exhibits some latency constraint which is also considerable. In comparison with OE and DyAD routing, our proposed algorithm performs much better than OE but is not able to give good results as DyAD routing in situations where all the nodes are loaded heavily in the network. We also have found that, the increasing number of VC buffers is not always able to give better performance as it can also increase the latency in the physical channel in a significant amount. In case of link BW utilization, our algorithm gives desirable results as all the nodes have a stable link BW utilization rate.

The proposed algorithm can be further modified to examine the performance of the network in case of fault tolerance and when there is a significant amount of traffic flow. Hardware implementation can also be possible but a runtime adaptation layer would be needed for mapping the different classes of traffic to different router port. Power overhead can't be predicted as power constraint is not considered here.

VIII. REFERENCES

[1] Ville Rantala, Teijo Lehtonen, Juha Plosila, "Network on Chip Routing Algorithms", University of Turku, Department of Information Technology, 2009

[2] Chowdhury, Zamshed Iqbal, et al. "Effect of technology scaling on leakage power consumption in on-chip switches." *Informatics, Electronics & Vision (ICIEV), 2014 International Conference on*. IEEE,2014.

[3] P. Pratim Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures", *IEEE Transactions on Computers*, vol. 54, no. 8, pp. 1025-1040, 2009.

[4] L. Benini and G. De Micheli. Networks on chip: A new paradigm for systems on chip design. In Proc. DATE, March 2002.

[5] Ankur Agarwal, Cyril Iskander, Ravi Shankar, "*Survey of Network on Chip (NoC) Architectures & Contributions*", Journal of Engineering, Computing & Architecture, Vol-03, Issue 1, 2009.

[6] Nayeen, Md Julkar, et al. "Performance evaluation of a portable PABX system through developing new bandwidth optimization technique." *Electrical Engineering and Information Communication Technology (ICEEICT), 2015 International Conference on*. IEEE, 2015.

[7] C. Nicopoulos *et al.*, "ViChaR: A dynamic virtual channel regulator for network-on-chip routers," in *Proc. MICRO*, 2014, pp. 333–346.

[8] Adaptive Network on Chip Routing using the Turn Model, by Jonathan W. Brown, University of New Hampshire, May 2013

[9] C. Nicopoulos *et al.*, "ViChaR: A dynamic virtual channel regulator for network-on-chip routers," in *Proc. MICRO*, 2006, pp. 333–346.

[10] A. Hansson *et al.*, "An on-chip interconnect and protocol stack for multiple communication paradigms and programming models," in *Proc.CODES+ISSS*, 2009, pp. 99–108.

[11] A. Pullini *et al.*, "Fault tolerance overhead in network-on-chip flow control schemes," in *Proc. SBCCI*, 2005, pp. 224–229.

[12] L. Benini *et al.*, "Networks on chips: A new SoC paradigm," *Computer*, vol. 35, no. 1, pp. 70–78, 2002.

[13] S. Rodrigo, "Efficient implementation of distributed routing algorithms for NoCs," in IET Computers & Digital Techniques, 2010.