A Buffer-aware Routing Algorithm for Irregular 2-D Network-on-Chip without Virtual Channel

Ladan Momeni
Department of Computer Engineering
Sama Technical and Vocational Training College
Islamic Azad University, Ahvaz Branch, Ahvaz, Iran
ladan.momeni@gmail.com

Arshin Rezazadeh
Department of Computer Engineering
Shahid Chamran University of Ahvaz
Ahvaz, Iran
a.rezazadeh@scu.ac.ir

Abstract — Network-on-Chips (NoCs) usually use regular mesh-based topologies. Regular mesh topologies are not always efficient because of power and area constraints which should be considered in designing system-on-chips. To overcome this problem, irregular mesh NoCs are used for which the design of routing algorithms is an important issue. This paper introduces a routing algorithm for irregular mesh-based NoCs called IIR for Mesh Networks. In contrast to other routing algorithms, this algorithm can be implemented on any arbitrary irregular mesh NoC and can tolerate solid irregular areas without using virtual channels. Moreover, the proposed algorithm can tolerate non-overlapping faulty regions. Furthermore, the proposed scheme misroutes messages both clockwise and counter clockwise directions to reduce channel contention on an oversized node (ON). The main idea of this algorithm borrowed from odd-even turn model and FT-Cube algorithm. Additionally, proposed algorithm is deadlock-free and livelock-free in meshes for non-overlapping multiple irregular areas and faulty regions in NoC interconnection network. IIR algorithm increases the throughput of physical links more than FTR with lower average message delay. As the simulation results show, IIR has a higher performance compared to FTR algorithm. The results also show that our algorithm has better performance with 100% traffic load in Network-On-Chip.

Keywords - Network-on-Chip; throughput; irregular routing algorithm; performance; deterministic routing

I. INTRODUCTION

Moore’s empirical law has been proved valid even in the nonmetric lithographic regime and we will likely continue to behold an exponential increase in silicon transistor counts in the years to come. As a corollary, we will see also an exponential increase in the “core” counts in chip-multiprocessors (CMP) for general purpose computing and in special purpose dedicated on-chip architectures like graphics processing units [1].

In recent years, with the development of Multi-Processor System-on-Chip (MPSoC), the intra-chip communication is becoming more and more important to the performance of the entire system. Recently, the Network-on-Chip (NoC) approach has been proposed as a promising technique to overcome this bottleneck. Each processing core is connected to a local router. All the routers, and the links interconnecting the routers, form the NoC. The communication between two processing cores in a NoC is very similar to that between two nodes in a computer network. As a result, a lot of existing research in computer networks has been leveraged for NoCs [2].

Network on Chip (NoC) design paradigm has been proposed as a viable communication solution for integrating large number of processing cores into a single chip [6].

The main components of NoC include IP cores, network interface (NI), routers and physical links. IP cores are some function units in the digital system, such as CPU, DSP, memory and I/O units. Network interface ensures the heterogeneous IP cores with different protocols communicating transparently. NI also conducts packet encapsulation, end-to-end flow control and packet reordering. Routers connected by the physical links are the most important components in the NoC architecture. They guarantee efficient data communication between any two IP cores in the system [3].

There is a large body of work on performance evaluation and design tradeoffs of multicore computer interconnection networks and NoC architectures. Although performance (e.g. network latency and throughput) is the key factor in designing multicore interconnection networks, in the design and implementation of NoC architectures, power consumption is the most important factor while performance is the second important measure [4].

The mesh structure is one of the most important interconnection network models. As a topology to interconnect multiprocessor computer systems, it has been shown to possess many attractive properties. It has been one of the most favored interconnection structures to manufacturers, and parallel computers using mesh or its variants have been commercially available for quite a long time [5].

This paper focuses on evaluating and comparing performance of two distinct types of adaptive deadlock free routing algorithms for irregular topology mesh networks. The rest of the paper is organized as follows. In Section 2 we review routing algorithm issues. Section 3 presents the previous routing algorithm and its modification on this algorithm, IIR algorithm. In Section 4, we present the evaluation, simulation conditions and results. Section 6
concludes the paper and lists some research problems for the future.

II. ROUTING ALGORITHM

Routing is a process to send messages, which can be either data or instructions, from a source node to a destination node, passing some intermediate nodes. There are basically two types of routing: deterministic routing and adaptive routing. In deterministic routing, a fixed path is used to send/receive messages for a particular pair of source/destination [5].

Routing algorithm decides the path that a packet should take to reach to its destination. Based on the adaptivity that the routing algorithms provide for packets, they can be divided into deterministic, partially adaptive, and fully adaptive categories. In the deterministic routing algorithms, packets have to pass through predefined paths i.e., they are not allowed to use all of possible paths toward their destinations. For example, dimension order routing algorithm restricts packets to pass through a subset of all possible paths. At the first phase of this routing algorithm, packets will be routed in X dimension till they reach the column of the destination node, then the second phase will start in which the packets are routed in Y dimension. During the second phase, packets are not allowed to turn to left or right [6].

Routing algorithm determines the route of each packet transmitting from its source to the destination. It is critical to the performance of Network-on-Chip. NoC routing algorithms can be divided into deterministic routing and adaptive routing. In the deterministic routing, the route of each packet is determined only by its source and destination addresses, while the adaptive routing algorithm chooses the path according to the network condition dynamically [3].

Deadlock is an anomalous network state in which a circular hold-and-wait dependency relation is formed among the network resources, causing packet routing to be indefinitely postponed. Meanwhile, in livelock situation, a packet travels continuously around the network without ever reaching its destination because the requested channels are constantly occupied by other packets. Minimal routing schemes use the shortest path from a source to a destination, thus the livelock problem can be avoided [7].

The region concept presented in was intended for use of larger resources, which do not fit in the fixed sized slot of a regular mesh architecture layout. Region concept could in addition be useful for encapsulating a group of resources which have very high and special communication requirements which cannot be supported by the general NoC communication infrastructure [8].

Efficient routing of messages within the network is essential in order to fully exploit the power of the computing resources and achieve good performance for applications running on them. A good routing algorithm should not only provide low latency for messages but should also be deadlock free when the network is concurrently routing multiple messages. However, incorporating regions in mesh networks result in a major change of the communication infrastructure and the existing mesh routing algorithms cannot be directly reused [8].

Wormhole routing is the preferred flow control strategy for application-specific topologies, providing the NoC components with low latency and buffering requirements. Although there are advantages to this type of routing, it is also prone to contention as the packets tend to spread through-out the network during transmission. Contention within a network occurs when different packets require the same resources at a particular moment in time. If a contention point propagates throughout the system, congestion is formed, causing performance degradation. As a result, the system can have long delays and may not meet throughput requirements as is necessary to adhere to its system demands [9].

III. IMPROVED IRREGULAR ROUTING ALGORITHM

This section describes how we evaluate an existing technique for fault-tolerant wormhole routing in NoC with a mesh-based topology to an improved irregular routing algorithm. The routing algorithm considered in this paper is a deterministic e-cube routing as long as no oversized nodes ahead. When facing an oversized node or link, a given flit cannot be routed along its normal e-cube route and its direction would be changed according to a bracket of rules and it would be re-routed along an oversized nodes or links. These rules have been put forward by Chalasani and Boppana [10]. The main idea is described in the rest of this section.

TABLE I. DIRECTION TO BE USED FOR MISROUTED MESSAGES ON OVERSIZED NODES

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Traversed on the oversized node</th>
<th>Position of Destination</th>
<th>Oversized node Orientation</th>
</tr>
</thead>
<tbody>
<tr>
<td>WE</td>
<td>No</td>
<td>In a row above its row of travel</td>
<td>Clockwise</td>
</tr>
<tr>
<td>WE</td>
<td>No</td>
<td>In a row below its row of travel</td>
<td>Counter Clockwise</td>
</tr>
<tr>
<td>WE</td>
<td>No</td>
<td>In the same row</td>
<td>Either orientation</td>
</tr>
<tr>
<td>NS or SN</td>
<td>No</td>
<td>Don't care</td>
<td>Either orientation</td>
</tr>
<tr>
<td>EW</td>
<td>No</td>
<td>In a row above its row of travel</td>
<td>Clockwise</td>
</tr>
<tr>
<td>EW</td>
<td>No</td>
<td>In a row below its row of travel</td>
<td>Counter Clockwise</td>
</tr>
<tr>
<td>EW</td>
<td>No</td>
<td>In the same row</td>
<td>Either orientation</td>
</tr>
<tr>
<td>Any message</td>
<td>Yes</td>
<td>Don't care</td>
<td>Choose the orientation that is being used by the message</td>
</tr>
</tbody>
</table>

A. Fault-Tolerant Routing (FTR) – Primitive Algorithm

The algorithm presented by Chalasani and Boppana, FTR, uses four virtual channels (VCs). This algorithm is able to pass faulty blocks and overlapped faulty regions. Each message is injected into the network as a row message.
(message must travel horizontally at first) and its status is set to normal. Messages are routed along their deterministic e-cube hop if they are not blocked by faults. When faults are happened, its status would be set to misrouted and depending on the message type and relative position of destination nodes to source nodes, direction of messages are set to clockwise or counter-clockwise by use of table 1 [10]. Messages are routed on border of faulty block according to specific directions. The status of a message which is passed the faulty region would be configured to normal again.

The technique evaluated in this paper has one primary advantage over the one presented in the previous work. According to [15], as long as no fault occurs, a flit always uses a fixed virtual channel (channel c0 for EW message, c1 for WE, c2 for NS and channel c3 used for SN messages). When oversized nodes are encountered and a flit is re-routed, it uses specific virtual channel depending on pre-defined set of rules. However, in the current paper, a flit is allowed to use all virtual channels instead of just one fixed virtual channel when its type is NS or SN and located on the boundary of an oversized node [14]. But in FTR algorithm using from just two virtual channels is permitted in this situation. Using this modification, simulations are performed to evaluate the performance of the enhanced algorithms compared to the algorithms proposed in prior work. Simulation results indicate an improvement in the average message delays and throughput of network for different fault rates and several traffic patterns. Furthermore, the enhanced approach can handle higher message injection rates, it means it has higher saturation rate in uniform, hotspot and local traffic. Moreover, throughput of evaluated algorithms showed that the new algorithm has higher performance in comparison with old one. Throughput illustrates the number of packets in each cycle which passed from a node [16].

C. Example

We now consider the example of routing message M from (3, 7) to (7, 3) in fig. 1. The path taken by M is also shown in fig. 1. M is routed as an EW message (row direction) when it is generated, depending on its direction of travel along the row. Once a message completes its row hops, it becomes a NS or a SN message (column direction) depending on its travel direction along the column. Thus, EW and WE messages will become NS or SN messages; however, NS and SN messages cannot change their types because of live-lock and dead-lock [14]. Evaluated irregular routing algorithm and procedures needed are given in algorithm 1 and algorithm 2.

Algorithm 1: Set-Message-Type and Set-Message Status procedures.

Algorithm 2: Improved-Irregular-Routing (IRR) procedure.

B. Improved Irregular Routing (IRR) - Modified Algorithm

The e-cube routes a message in a row until the message reaches a node that is in the same column as its destination, and then routes it in the column. For regular meshes, the e-cube provides deadlock-free shortest path routing without requiring multiple virtual channels to be simulated. At each point during the routing of a message, the e-cube specifies the next hop which should be taken by the message. The message is assumed to be blocked by an oversized node, if its e-cube hop is on an oversized node or link [11] [12] [13]. The evaluated modification uses number of VCs as same as primitive algorithm. An entire column/row oversized node which disconnects meshes has not been considered [14].

To route messages around oversized nodes, they are classified into one of the following types: EW (East-to-West), WE (West-to-East), NS (North-to-South), or SN (South-to-North). A message is labeled as either an EW or WE message (row direction) when it is generated, depending on its direction of travel along the row. Once a message completes its row hops, it becomes a NS or a SN message (column direction) depending on its travel direction along the column. Thus, EW and WE messages will become NS or SN messages; however, NS and SN messages cannot change their types because of live-lock and dead-lock [14]. Evaluated irregular routing algorithm and procedures needed are given in algorithm 1 and algorithm 2.

The technique evaluated in this paper has one primary advantage over the one presented in the previous work. According to [15], as long as no fault occurs, a flit always uses a fixed virtual channel (channel c0 for EW message, c1 for WE, c2 for NS and channel c3 used for SN messages). When oversized nodes are encountered and a flit is re-routed, it uses specific virtual channel depending on pre-defined set of rules. However, in the current paper, a flit is allowed to use all virtual channels instead of just one fixed virtual channel when its type is NS or SN and located on the boundary of an oversized node [14]. But in FTR algorithm using from just two virtual channels is permitted in this situation. Using this modification, simulations are performed to evaluate the performance of the enhanced algorithms compared to the algorithms proposed in prior work. Simulation results indicate an improvement in the average message delays and throughput of network for different fault rates and several traffic patterns. Furthermore, the enhanced approach can handle higher message injection rates, it means it has higher saturation rate in uniform, hotspot and local traffic. Moreover, throughput of evaluated algorithms showed that the new algorithm has higher performance in comparison with old one. Throughput illustrates the number of packets in each cycle which passed from a node [16].

C. Example

We now consider the example of routing message M from (3, 7) to (7, 3) in fig. 1. The path taken by M is also shown in fig. 1. M is routed as an EW message from (3, 7) to (3, 6). At (3, 6), its next e-cube hop is an oversized node and its direction is set to clockwise. At (4, 6), its direction is reset to normal and M is routed along its e-cube hop to (4, 3). At (4, 3), M becomes an NS message and travels from (4, 3) to (5, 3). At node (5, 3), due to the fact that its next e-cube hop is an oversized node, M travels in the clockwise direction to
Direction of M is reset to normal again at (6, 3) and M is routed along its e-cube hop to destination node (7, 3). Step 2 of the path use virtual channel vc0, steps 6 and 8 use vc2 according to previous notes. Steps 1, 3, 4, 5, 6, 8 and 10 all four virtual channels can be used and demonstrated by solid arrows.

IV. RESULTS AND DISCUSSIONS

In this section, we will describe how we perform the simulation and acquire results from simulator. On-chip network is by far resources limited, e.g. limited by wiring, buffer and area. In the standard computer networks, performance is drastically impacted by the amount of buffering resources available, especially when the traffic is congested. In contrast, on-chip networks should use the least amount of buffer space due to the limited energy and area budget [17]. Network throughput can be increased by organizing the flit buffers associated with each physical channel into several virtual channels. A virtual channel consists of a buffer, together with associated state information, capable of holding one or more flits of a message [18]. Furthermore, we show the enhancements of the improved algorithm.

Figure 1. Example of routing message from (3,7) to (7,3). ON1, ON2 and ON3 are oversized nodes.

A. Simulation Structure

In order to model the interconnection network, an object-oriented simulator was developed base on [19] [20] [21]. The simulator is structured so that classes, such as routing algorithm or message traffic can be changed with-out any change in other components. A flit-level simulator has been designed. We record average message latencies measured in the network with the time unit equal to the transmission time of a single flit (one clock cycle). Our generation interval has exponential distribution which leads to Poisson distribution of number of generated messages per a specified interval. Our study is performed for 10% oversized nodes rates. In our simulation studies, we assumed message length to be equal to 32 flits and we used an 8 x 8 2-D mesh network. In each case, we have randomly generated the required number of oversized nodes and links. Three traffic patterns are simulated:

- Uniform - The source node sends messages to any other node with equal probability.
- Local - The source node sends messages to any other node with equal probability but with fixed maximum distance. We use Manhattan distance calculated as follows:

\[ D_m = |x_s - x_d| + |y_s - y_d| \] (1)

In equation above, D means distance and m denotes to Manhattan. Furthermore, x denotes to dimension x and y denotes to dimension y. Likewise, s used for source node and d is destination node.
- Hotspot - Messages are destined to a specific node with a certain probability and are otherwise uniformly distributed.

Uniform traffic is the most pattern which have been used for interconnection networks of NoC. As the mesh interconnection network is not a symmetric network, we have considered two types of simulation for hotspot traffic in this network. In one group of simulations, a corner node is selected as the hotspot node and in the other group; a node in the middle of the network is chosen as the hotspot node, and finally averaged. Hotspot rate is also considered in our study, namely 10% [22].

Figure 2. Throughput of FTR and IIR in hotspot traffic

Figure 3. Throughput of FTR and IIR in local traffic

We also considered local traffic for this comparison and to show the effect of mapping of elements. As learned by simulation results for uniform traffic at our study, it is...
founded that average number of hopes for this 8 × 8 mesh with this algorithm and other similar algorithms such as if-cube3 [23] is about 5. As the result of this practice, we used 5 Manhattan distance for messages to learn if any node needs no more than 5 hopes for destination what happened.

The number of messages generated for each simulation result depends on the network size and traffic distribution. It is in the range of 3 to 4 million messages. The simulator has three phases: start-up, steady-state, and termination. The start-up phase has been used to ensure that network is in steady-state before measuring message latency and other parameters. For this reason the statistics for the first 10% of generated messages have not been gathered [19]. All measurements are obtained from the remaining of messages generated in steady-state phase. The termination phase would continue till all the messages generated are delivered.

In the remaining part of this section, we described the effect of using modified algorithm, IIR, on the performance of deterministic routing in the mesh network in details.

### B. Uniform, Local and Hotspot Traffic Patterns

We defined throughput as the major performance metric. For an interconnect network, the system designer will specify a throughput requirement. Figures 2 to 4 show the simulation results for three different traffic patterns, hotspot, local and uniform pattern with 32 flits on 8 × 8 mesh. Fig. 2 shows the throughput over the message injection rate for two mentioned algorithms. As we can see, the throughput of the network which uses FTR algorithm is lower while the IIR algorithm has higher network throughput. As an example in FTR algorithm, the throughput for 0.001 message injection rate (MIR) is 11.18% at 100% traffic load; however, the other algorithm, IIR, could achieve 17.19% throughput in 0.0015 MIR at high traffic load – more than 53% improvement in this case. In fact our irregular routing algorithm has higher throughput for higher MIRs.

Fig. 3 shows throughput over MIR in local traffic. As we can see, the performance of IIR is higher than FTR algorithm and also has higher saturation point. Furthermore, for uniform traffic, it is showed in fig. 4 which IIR algorithm has higher performance in this traffic. It is clear that in all three traffic patterns IIR algorithm has higher throughput and higher saturation point compared to FTR algorithm.

The most valuable comparison we have done between these two algorithms is the rate of average message delay over throughput. Comparative performance across different traffic patterns in figures 5, 6 and 7 is specific to the several oversized node sets used. For each case, we have simulated previous fault sets for 100% traffic load. The injection control helps us here; otherwise, we would have to perform the tedious task of determining the saturation point for each fault set. As an example in fig. 5, we can look at the amount of average message delay for both algorithms with 11% throughputs. In this point of throughput, the network which is using FTR has more than 103 AMD at 100% traffic load while the other network, using IIR, has less than 52 AMD, and it has not been saturated. Comparing the throughput of these algorithms for 100% traffic load, it is obvious the network using IIR has 17% throughputs whereas the other one has just 11% throughputs. We have improved throughput of network more than 54% by our evaluated algorithm at 100% traffic load.
important parameters to consider in irregular routing interconnection network and power consumption, are messages passed the oversized regions. Both delay of mesh size, and traffic patterns. All of parameters we have both FTR and IIR algorithms for the same message injection better by using the same virtual channels. We have simulated results using IIR; our algorithm, called IIR, however, works physical channel are needed in the evaluated algorithm. There is no restriction on the algorithm is simple, easy and its principle is similar to the previous algorithm, FTR. There is no restriction on the number of oversized nodes and similar virtual channels per physical channel are needed in the evaluated algorithm.

We also showed that in different traffic patterns these oversized nodes can be handled. The deterministic algorithm is enhanced from the non-adaptive counterpart by utilizing the virtual channels that are not used in the non-faulty conditions. The method we used for enhancing the IIR algorithm is simple, easy and its principle is similar to the previous algorithm, FTR. There is no restriction on the number of oversized nodes and similar virtual channels per physical channel are needed in the evaluated algorithm.

We presented that the studied parameters have acceptable results using IIR; our algorithm, called IIR, however, better by using the same virtual channels. We have simulated both FTR and IIR algorithms for the same message injection rates, oversized nodes situations, message lengths, network size, and traffic patterns. All of parameters we have examined show better results for IIR in comparison with FTR algorithm.

We could achieve higher performance by modifying deterministic algorithms, especially focusing on how messages passed the oversized regions. Both delay of mesh network by different message injection rates, utilization of interconnection network and power consumption, are important parameters to consider in irregular routing algorithms for Network-On-Chips.

V. CONCLUSION

Designing a deadlock-free routing algorithm that can tolerate unlimited number of oversized nodes is not an easy task. The simulation results show that up to 50% improvement of network throughput, which is needed to work with rectangular oversized nodes, can be recovered if oversized nodes and links is less than 10% of the total network.

We also showed that in different traffic patterns these oversized nodes can be handled. The deterministic algorithm is enhanced from the non-adaptive counterpart by utilizing the virtual channels that are not used in the non-faulty conditions. The method we used for enhancing the IIR algorithm is simple, easy and its principle is similar to the previous algorithm, FTR. There is no restriction on the number of oversized nodes and similar virtual channels per physical channel are needed in the evaluated algorithm.

We presented that the studied parameters have acceptable results using IIR; our algorithm, called IIR, however, better by using the same virtual channels. We have simulated both FTR and IIR algorithms for the same message injection rates, oversized nodes situations, message lengths, network size, and traffic patterns. All of parameters we have examined show better results for IIR in comparison with FTR algorithm.

We could achieve higher performance by modifying deterministic algorithms, especially focusing on how messages passed the oversized regions. Both delay of mesh network by different message injection rates, utilization of interconnection network and power consumption, are important parameters to consider in irregular routing algorithms for Network-On-Chips.

REFERENCES


