INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES
&
MANAGEMENT

IMPLEMENTATION OF TWO PHASE X-Y ROUTING IN NOC ROUTER

Rajesh Nema*[1], Parvinder Singh Khanna[2]

Abstract

Power and performance play a significant role since the size of technology to build modern digital systems are reduced. Therefore, in designing these systems, all of the designing features shall somehow acquire their confirmation from the standpoint of these two parameters. One of the important features is communication. Communication portion in the power consumption of System on Chip can be up to 50% of the whole power consumption of the chip. This deems to be more important for Network on Chips which center around an intercommunication networks. In this article, designing and implementation a NoC router based on two phase x-y routing algorithm is presented.

Keywords:-NOC, Two-phase routing.

Introduction

Network-on-Chip or Network-on-a-Chip (NoC or NOC) is an approach to designing the communication subsystem between IP cores in a System-on-a-Chip (SoC). NoCs can span synchronous and asynchronous clock domains or use unclocked asynchronous logic. NoC applies networking theory and methods to on-chip communication and brings notable improvements over conventional bus and crossbar interconnections. NoC improves the scalability of SoCs, and the power efficiency of complex SoCs compared to other designs. Network-on-Chip (NoC) is an emerging paradigm for communications within large VLSI systems implemented on a single silicon chip. In a NoC system, modules such as processor cores, memories and specialized IP blocks exchange data using a network as a "public transportation" sub-system for the information traffic. A NoC is constructed from multiple point-to-point data links interconnected by switches (routers), such that messages can be relayed from any source module to any destination module over several links, by making routing decisions at the switches.

*** Corresponding Author**
E. mail: rajeshnema2010@rediffmail.com

A NoC is similar to a modern telecommunications network, using digital bit-packet switching over multiplexed links. Although packet-switching is sometimes claimed as necessity for a NoC, there are several NoC proposals utilizing circuit-switching techniques. This definition based on routers is usually interpreted so that a single shared bus, a single crossbar switch or a point-to-point network are not NoCs but practically all other topologies are. This is somewhat confusing since all above mentioned are networks (they enable communication between two or more devices) but they are not considered as network-on-chips. Note that some articles erroneously use NoC as a synonym for mesh topology although NoC paradigm does not dictate the topology. Likewise, the regularity of topology is sometimes considered as a requirement which is, obviously, not the case in research concentrating on "application-specific NoC topology synthesis". The wires in the links of the NoC are shared by many signals. A high level of parallelism is achieved, because all links in the NoC can operate simultaneously on different data packets. Therefore, as the complexity of integrated systems keeps growing, a NoC provides enhanced performance (such as throughput) and scalability in comparison with previous communication architectures (e.g., dedicated point-to-point signal wires, shared buses, or segmented buses with bridges). Of course, the algorithms must be designed in such a way that they offer large parallelism and can hence utilize the potential of NoC.. The trend toward faster and lower power communication may decrease reliability as an

unfortunate side effect. In NoC design approach, physical layer design finds a compromise between competing quality metrics and provide a clean and complete abstraction of channel characteristics to above micro-network layers [1]. The data-link layer abstracts the physical layer as an unreliable digital link. The main purpose of data-link protocols is to increase the reliability of the link up to a minimum required level, under the assumption that the physical layer by itself is not sufficiently reliable [2]. At the data-link layer, a key task is to achieve the specified communication reliability level [3]. At the network layer, packet data transmission can be customized by the choice of switching and routing algorithms. The former, (e.g., circuit, packet, and cut-through switching), establishes the type of connection while the latter determine the path followed by a message through the network to its final destination. Switching and routing algorithms for NoC affect heavily performance. Atop the network layer, the communication abstraction is an end-to-end connection. The transport layer decomposes messages into packets at the source. It also decides the order of packets and reassembles the messages at
the destination. Another transport-layer task is flow control. Flow control can mitigate the effect of congestion by regulating the amount of data that enters the network, at the price of some throughput penalty [4].

Routing Algorithm

The routing algorithm, which defines the path taken by a packet between the source and the destination, is a main task in network layer design of NoC. According to where routing decisions are taken, it is possible to classify the routing in source and distributed routing [5].

In source routing, the whole path is decided at the source router, while in distributed routing each router receives a packet and decides the direction to send it. According to how a path is defined to transmit packets, routing can be classified as deterministic or adaptive. In deterministic routing, the path is uniquely defined by the source and target addresses. In adaptive routing, the path is a function of the network traffic [5][6].

Two-Phase Routing

In the area of fault tolerant routing, routing algorithms based on two-phase routing scheme have been presented [10][11]. The scheme avoids faulty nodes by selecting an intermediate node.

A. Basic ideas

Since irregular mesh with oversized cells is used to connect various modules of application-specific NoCs, certain nodes are not able to reach other nodes by the available routing algorithms. So we present a two-phase routing scheme to solve the issue. The presented scheme uses available routing algorithms as basic routing algorithms. It is in fact a combination of two basic routing algorithms. The routing path from source node to a destination node is split into two phases. In the first phase, messages are routed from a source node to an intermediate node, and then from
the intermediate node to a destination node in the second phase. For example, as shown in figure 1, a message uses XY routing algorithm to routes from a source node ‘a’ to a intermediate node ‘c’ in the first phase, and then also uses X-Y routing algorithm to routes from the intermediate node ‘c’ to a destination node ‘d’.

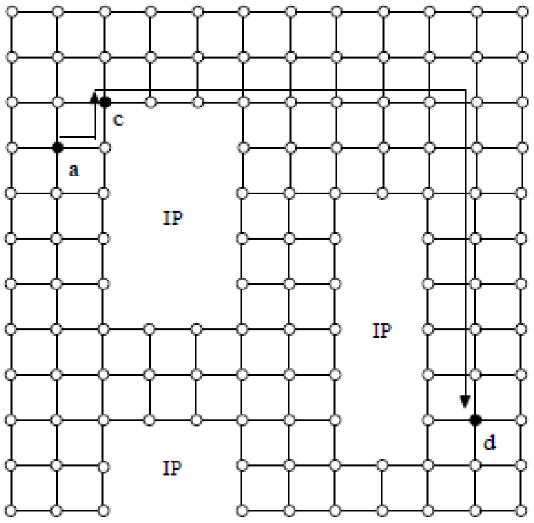


Figure.1: A irregular mesh with 3 oversized IP modules

It is assumed that the node ‘c’ do not store the whole message after a message reached it at the end of the first routing phase. It simply forwards the flits of the message in a pipelined fashion. Since a message can reserve a sequence of nodes, it is possible that the head flit of a message reserve nodes between ‘c’ and ‘d’, while its tail flit still reserve nodes between ‘a’ and ‘c’. Furthermore, if we use different virtual channel during each of the routing phase, the routing algorithm is guaranteed to be deadlock-free.

B. XY Routing Algorithm

The XY routing algorithm is one kind of distributed deterministic routing algorithms. For a 2-Dimesion mesh topology NoC, each router can be identified by its coordinate (x, y) (Fig. 2). The XY routing algorithm compares the current router address (Cx, Cy) to the destination router address (Dx, Dy) of the packet, stored in the header flit. Flits must be

routed to the core port of the router when the (Cx, Cy) address of the current router is equal to the (Dx, Dy) address. If this is not the case, the Dx address is firstly compared to the Cx (horizontal) address. Flits will be routed to the East port when Cx<Dx, to West when Cx>Dx and if Cx=Dx the header flit is already horizontally aligned. If this last condition is true, the Dy (vertical) address is compared to the Cy address. Flits will be routed to South when Cy<Dy, to North when Cy>Dy. If the chosen port is busy, the header flit as well as all subsequent flits of this packet will be blocked.
The routing request for this packet will remain active until a connection is established in some future execution of the procedure in this router. The following text is the algorithm XY:

```
Algorithm XY
/*Source router: (Sx,Sy);destination router:
(Dx,Dy); current
router: (Cx,Cy).*/
begin
if (Dx>Cx) //eastbound messages
return E;
else
if (Dx<Cx) //westbound messages
return W;
else
if (Dx=Cx) { //currently in the same column as
//destination
if (Dy<Cy) //southbound messages
return S;
else
if (Dy>Cy) //northbound messages
return N;
else
if (Dy=Cy) //current router is the destination router
return C;
}
End
```

C. *Two-Phase X-Y Routing Algorithm*

In this section, we introduce a new fault-tolerant routing algorithm based on the two-phase scheme. The presented fault tolerant routing algorithm denoted as two-phase X-Y routing algorithm is described as follows, where xc, yc represent x, y coordinates of the current node.
xm, ym represent x, y coordinates of the intermediate node. xd, yd represent x, y coordinates of the destination node. We use one bit “current_phase” to indicate if a message uses the first phase (0) or the second phase (1) to route. xoffs, yoffs represent the offsets of x, y coordinate between the current node and the intermediate node (current_phase=0), or between the current node and the destination node (current_phase=1). Each physical channel of x or y dimension is split into 2 virtual channels –X0, +X1 or –Y0, +Y1.

```
Algorithm two-phase X-Y routing (xc,yc):
begin
if current_phase=0 then
xoffs←xm – xc
yoffs←ym – yc
if xoffs=0 and yoffs=0 then
current_phase=1
end if
end if
if current_phase=1 then
xoffs←xd – xc
yoffs←yd – yc
if xoffs=0 and yoffs=0 then
chn←internal
return
end if
end if
if xoffs<0 then
chn←-Xcurrent_phase
end if
if xoffs>0 then
chn←+Xcurrent_phase
end if
if xoffs=0 and yoffs<0 then
chn←-Ycurrent_phase
end if
if xoffs=0 and yoffs>0 then
chn←+Ycurrent_phase
end if
end
```

According to the two-phase X-Y routing algorithm for irregular mesh-based NoCs, messages round oversized components through two phases. Messages are typed as the first phase message (current_phase=0) or the second phase message (current_phase=1). Each message is initially the first phase message (0). The first phase messages are routed through the first basic X-Y routing function. The first phase messages will not become the second phase messages until the offsets between the current node and the intermediate node equal to 0. The second phase messages are routed through the second basic X-Y routing function until they arrive at their destinations.

Theorem: *Two-Phase X-Y* algorithm is deadlock-free.
Proof:
The channel class dependency graph of basic X-Y algorithm Dcc=G(CC, Ecc) is shown in Figure 2(a), where verticals represent the channel classes -x, +x, -y and +y.

Directed arcs between channel classes represent that message can be routed from one channel class to another. The channel class dependency graph of two-phase X-Y algorithm Dcc=G(CC, Ecc) is shown in Figure 2(b). All the strongly connected branch of channel class dependency graph Dcc is shown in Figure 2(c).The channel classes of 8 strongly connected branch of Dcc are respectively CC1={-x0}, CC2={+x0}, CC3={-y0}, CC4={+y0}, CC5={-x1}, CC6={+x1}, CC7={-y1} and CC8={+y1}. The directions contained by them are DR(CC1)=DR(CC5)={-x}, DR(CC2)=DR(CC6)={+x}, DR(CC3)= DR(CC7)={-y} and DR(CC4)= DR(CC8)={+y}. ∀a strongly connected branch Dcci=G(CCi, Ecci) such that {+x, -x, +y, -y}≠DR(CCi), where i=1,

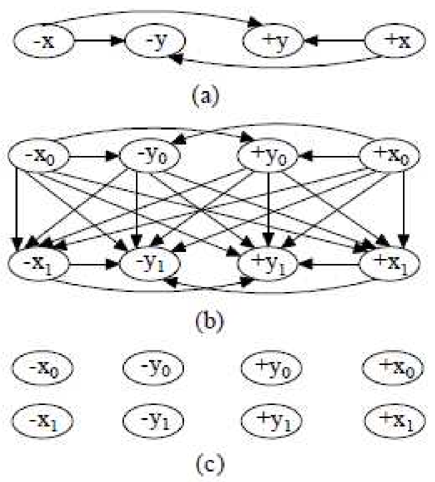


Figure.2: The channel class dependency graph of two-phase X-Y

D. Selection of the intermediate nodes

The original X-Y routing algorithm may not keep its connection because of regions which act as obstacles to the network traffic. More nodes of irregular mesh may be reached through the proposed two-phase X-Y routing algorithm. The two-phase X-Y routing algorithm selects an intermediate node between a source node and a destination node. Before we show how to find the intermediate nodes between any pairs of nodes, we first introduce some definitions and formula.

Definition.1: The rectangular area of mesh is called a zone of (x1,y1,x2,y2), where x1,y1 represent the upper left corner coordinates of the rectangular area and x2,y2 represent the lower right corner coordinates of it. The whole mesh of n×n can also be called a zone of (0, 0, n-1, n-1).

Definition.2: Given a irregular mesh of n × n with rectangular regions, for each region of (x1, y1, x2, y2), there is 4 pairs of zones for each region (R0, C0), (R1, C1), (R2, C2) and (R3, C3), where R0 is a zone of (0, y1+1, x1, y2-1), C0 is a zone of (x1+1, 0, n-1, n-1), R1 is a zone of (x2, y1+1, n-1, y2-1), C1 is a zone of (0, 0, x2-1, n-1), R2 is a zone of (0, y1+1, n-1, n-1), C2 is a zone of (x1+1, 0, x2-1, y1), R3 is a zone of (0, 0, n-1, y2-1) and C3 is a zone of (x1+1, y, x2-1, n-1).

For example, figure 3 shows a mesh of 6×6 and a region of (1, 1, 4, 4). There are 4 pairs of zones which are denoted as (R0, C0), (R1, C1), (R2, C2) and (R3, C3). As shown in figure 3, R0 is a zone of (0, 2, 3, 3), C0 is a zone of (2, 0, 5, 5), R1 is a zone of (4, 2, 5, 3), C1 is a zone of (0, 0, 3, 5), R2 is a zone of (0, 2, 5, 5), C2 is a zone of (2, 0, 3, 1), R3 is a zone of (0, 0, 5, 3) and C3 is a zone of (2, 4, 3, 5).

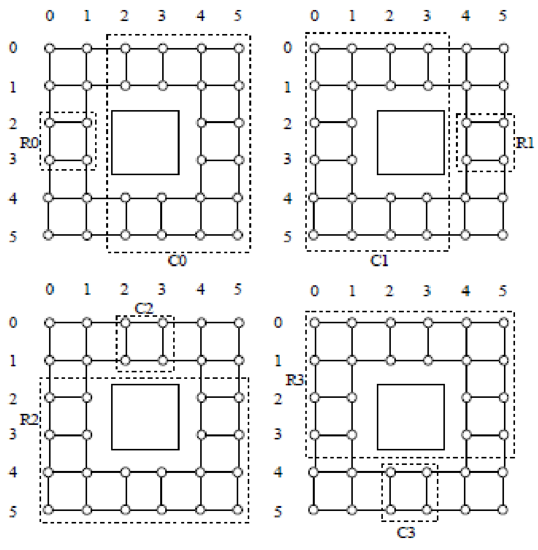


Figure.3: Eight X-zones of irregular mesh with three IP modules

It is easy to see that messages from the zones R0, R1, R2 or R3 cannot reach nodes in the zones C0, C1, C2 or C3 respectively through the original X-Y routing algorithm, since the region is larger than a tile of mesh. In order to route messages around the region, the proposed two-phase X-Y routing algorithm routes messages from the source node to a intermediate node in the first phase, then from the intermediate node to the destination node in the second phase.

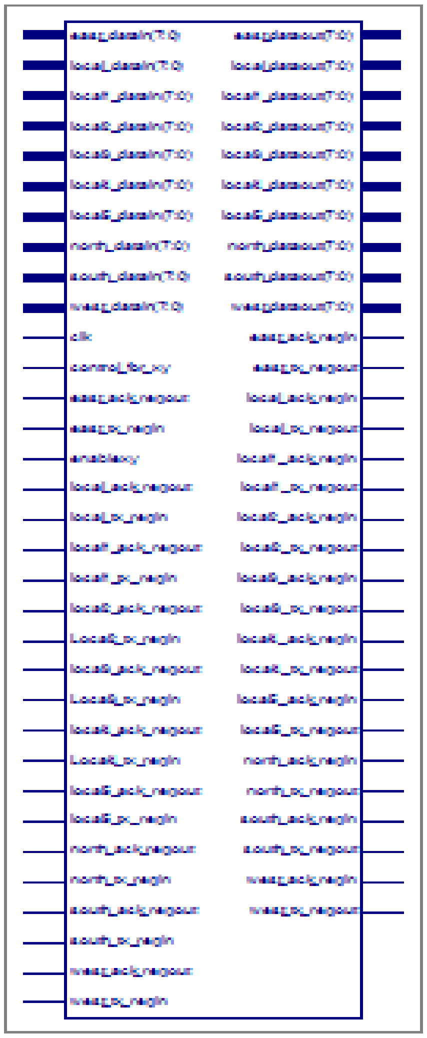
So the first step is to find a intermediate node for each pair of node between Ri and Ci shown in figure 3 before the two-phase X-Y routing algorithm can work, where i=0,…,3. We use the following approach.

Step 1: Calculate the number of 1-round hops for each pair of nodes. If the two nodes is connected through 1- round X-Y routing algorithm, the number of 1-round hops between them equals to the their total offset of dimension X and Y, otherwise, it equal to ∞.

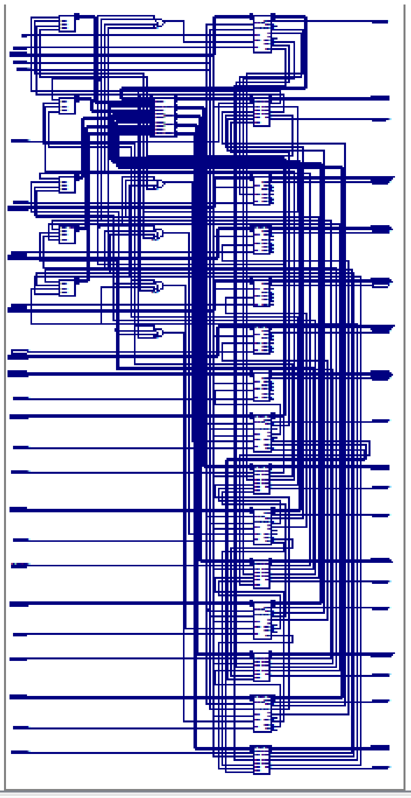
Step 2: Calculate the number of two-phase hops for each pair of nodes through each intermediate node. The number of two-phase hops between the source node and destination node through a intermediate node equal to the number of hops between the source node and intermediate node, plus the number of hops between the intermediate node and destination node.

Step 3: For each pair of nodes, select a node as the intermediate node of them through which the number
of hops between them is minimum.

Result



Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1429	6144	23%
Number of Slice Flip Flops	1655	12288	13%
Number of 4 input LUTs	1461	12288	11%
Number of bonded IOBs	193	240	80%
Number of FIFO16/RAMB16s	5	48	10%
Number of GCLKs	1	32	3%



Conclusion

Design has been tested for mesh network NoC router. It is implemented in Xilinx 13.1 ise on vertex xcv2-256 and its simulated with modelsim. We have justified two dimensional deadlock free operations in the proposed architecture. Of course latency is the issue in design because it follows store and forward mechanism.

References

1. A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. Oberg, M. Millberg.etal,“Network on a chip: an architecture for billion transistor era,” Proc. IEEE NorChip, 2000.
2. W. Dally and B. Towles, “Route packets, not wires: on-chip interconnection networks,” Proc. Design Automation Conference, Jun. 2001, pp. 684–689
3. S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Oberg . et al, “A network on chip architecture and design methodology,” Proc. IEEE Computer Society Annual Symposium on VLSI, Apr. 2002, pp.105–112, doi: 10.1109/ISVLSI.2002.1016885
4. R. Marculescu, H. Jingcao, U. Y. Ogras, “Key research problems in NoC design: a holistic perspective,” Proc. Third IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS’05) , Sept. 2005, pp.69-74, doi:10.1145/1084834.1084856
5. J. Duato, S. Yalamanchili, L. Ni, Interconnection networks: an engineering approach, Morgan Kaufmann, Revised Edition, 2002.
6. L. M. Ni, P. K. Mckinley, “A Survey of Wormhole Routing Techniques in Direct Networks”, Computer, vol. 26(2), Feb 1993, pp.62 –76, doi: 10.1109/2.191995