Journal of Systems Architecture 57 (2011) 109–120

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

A generic adaptive path-based routing method for MPSoCs

Masoud Daneshtalab *, Masoumeh Ebrahimi, Thomas Canhao Xu, Pasi Liljeberg, Hannu Tenhunen

Department of Information Technology, University of Turku, Finland

ARTICLE INFO

Article history:
Received 29 March 2010
Received in revised form 26 July 2010
Accepted 9 August 2010
Available online 18 August 2010

Keywords:
Network-on-Chip
Unicast and multicast routing algorithms
Adaptive routing algorithms
Hamiltonian path-based routing algorithms

ABSTRACT

Several unicast routing protocols have been presented for unicast traffic in MPSoCs. Exploiting the unicast routing algorithms for multicast traffic increases the likelihood of deadlock and congestion. In order to avoid deadlock for multicast traffic, the Hamiltonian path strategy was introduced. The traditional Hamiltonian path routing protocols supporting both unicast and multicast traffic are based on deterministic models, leading to lower performance. In this paper, we propose an adaptive routing protocol for both unicast and multicast traffic without using virtual channels. The proposed method maximizes the degree of adaptiveness of the routing functions which are based on the Hamiltonian path while guaranteeing deadlock freedom. Furthermore, both unicast and multicast aspects of the presented method have been widely investigated separately. Results obtained in both synthetic and real traffic models show that the proposed adaptive method for multicast and unicast aspects has lower latency and power dissipation compared to previously proposed path-based multicasting algorithms with negligible hardware overhead.

1. Introduction

As the technology scaling allows dozens or hundreds of processing elements to be integrated on a single chip, the interconnection between processing elements become more and more complicated and inefficient with traditional bus-based Multi-Processor System-on-Chip (MPSoC) architectures [1,2]. Recently many researchers have focused on the communication structures of MPSoCs to improve the scalability, power efficiency, and communication latency. Network-on-Chip (NoC) has emerged as a solution for the communication in complex MPSoCs [1,3,4] due to its reusability, scalability, and parallelism.

The choice of routing protocols can have a large impact on performance, latency and power consumption. The routing protocols in NoCs and MPSoCs can be either unicast (one-to-one) or multicast (one-to-many) [5,6]. In the unicast communication a message is sent from a source node to a single destination node, while in the multicast communication a message is delivered from one source node to an arbitrary number of destination nodes. Recently, multicast protocol is frequently used in network-based MPSoCs for many parallel applications such as cache coherency in distributed shared-memory architectures [7], clock synchronization [8], replication [9], and barrier synchronization [10]. Using unicast routing algorithms for multicast traffic increases the probability of deadlock. To avoid deadlock, additional hardware resources are required to support multicast traffic [11,12].

In this work, we present an adaptive method in a wormhole router network for both unicast and multicast traffic in two-dimensional (2D) mesh NoCs. The proposed Hamiltonian Adaptive Multicast Unicast Method (HAMUM) is based on the Hamiltonian path [13] such that the method is deadlock free. The presented method brings adaptivity to both multicast and unicast aspects of Hamiltonian path routing algorithms. Experimental results across a variety of synthetic and real application workloads show that power and performance characteristics are improved by exploiting HAMUM, and the chip area overhead of this scheme is less than 0.5%. The rest of the paper is organized as follows. The related work is discussed in Section 2. In Section 3, the motivation of the paper is described while a brief review of the Hamiltonian path and traditional path-based multicast models is given in Section 4. The proposed adaptive method and unicast/multicast aspects of this method are introduced in Section 5. The hardware implementation and results are presented in Sections 6 and 7, respectively, with the summary and conclusion given in the last section.

2. Related work

Multicast routing algorithms can be classified as unicast-based [14,15], tree-based [14,16], and path-based [6,17]. In the unicast-based, the multicast operation is performed by sending a separate copy of a message from a source to every destination or, alternatively, by sending unicast messages to subset of destinations. The drawback of this scheme is the fact that multiple copies of the same message are injected into the network, leading to increased

* Corresponding author. Tel.: +358 23336941.
E-mail address: masdan@utu.fi (M. Daneshtalab).

1383-7621/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.sysarc.2010.08.002

traffic in the network. Furthermore, each copy of the message loses considerable startup latency at the source. In the tree-based multicast approach, a spanning tree is constructed, so the source is indicated as the root and messages are sent down the tree. In this way a message might be replicated at some of the intermediate nodes and forwarded along multiple outgoing channels toward disjoint subsets of destinations. If one branch of the tree is blocked, all are blocked. Branches must proceed forward in lock step [18], which may cause a message to hold many channels for an extended period, thereby increasing the network contention [13]. The tree-based algorithms are efficient in the low injection rates, however, in the high injection rates or workloads near saturation the path-based algorithms are more efficient [14]. A tree-based multicast routing algorithms [14] has been proposed for on-chip interconnection network to overcome the tree-based drawbacks. The complexity, and hence, the hardware overhead of the presented method hardly depended to the network size. Besides, for updating routing tables, discrete unicast setup messages per destination should be sent by the source node. If the number of destinations grows, the number of unicast setup message will be increased, thereby reducing the performance. A solution to overcome the tree-based disadvantages is to utilize the path-based multicast wormhole routing. In this method, a source node prepares a message for delivery to a set of destinations by first sorting the addresses of the destinations in the correct order in which they are visited in the path, and then placing this sorted list in the header of the message. When the header entered a router with address A, the router checked to see if A is the next address in the header. If so, the address A is removed from the message header and a copy of data flits will be forwarded both to the local core and the next node on the path. Otherwise, the message is forwarded only to the next node on the path. In this way, the message is eventually delivered to every destination specified in the header. A number of studies have shown that a path-based approach exhibits superior performance characteristic over their unicast-based and tree-based counterparts [13,19].

To improve the path-based method, we propose an adaptive, deadlock-free method which can bring adaptivity for all of Hamiltonian based models. In fact, unlike other adaptive models in communication networks which are applicable only for unicast traffic, the proposed method handles both unicast and multicast traffic adaptively.

3. Motivation

Several deterministic and adaptive routing algorithms such as XY [20], Odd–Even [20], DyAD [21], etc. were proposed for unicast traffic in NoCs. These algorithms are deadlock free and do not require virtual channels to avoid deadlock. However, utilizing unicast routing algorithms for multicast traffic increases the likelihood of deadlock in the network. The following example shows the occurrence of deadlock when using the XY routing algorithm for multicast traffic. Consider a 2D-mesh (see Fig. 1) with bidirectional channels and suppose two multicast messages are generated in the network: $m1 = (7, \{10, 15\})$ and $m2 = (11, \{6, 3\})$. When the message m1 arrived to the corresponding local core (10), the message should be routed to the next destination address, 15. While the message $m1$ has acquired channels (7, 6), (6, 5), and (5, 10), it requires channel (10, 9), however, the channel (10, 9) has been acquired by message $m2$. The message $m2$ also acquired channels (11, 10), (10, 9) and (9, 6) and waiting for channel (6, 5). Due to the fact that messages cannot advance toward their destinations, the deadlock occurs. This example shows that an extension of the traditional unicast routing algorithms to a multicast routing can lead to deadlock.

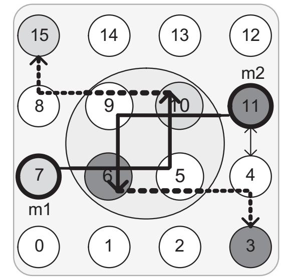

Fig. 1. Extension of XY method to multicast packets leads to deadlock.

On the other hand, several path-based multicast routing algorithms based on the Hamiltonian path were proposed to guarantee deadlock freedom [6,13]. But the traditional path-based algorithms are deterministic for both unicast and multicast traffic which degrades the performance significantly. This motivated us to propose a path-based method to bring adaptivity for both unicast and multicast traffic without using virtual channels.

4. Hamiltonian path strategy

Formally, an $m \times n$ 2D-mesh consists of $N = m \times n$ nodes; each node has an associated integer coordinate pair (x, y), $0 \leqslant x < n$ and $0 \leqslant y < m$. Two nodes with coordinates (x_i, y_i) and (x_j, y_j) are connected by a communication channel if and only if $|x_i - x_j| + |y_i - y_j| = 1$.

The path-based routing algorithm is established as the Hamiltonian path strategy. In this method an undirected Hamiltonian path of the network is constructed; the Hamiltonian path visits every node in a graph exactly once [22]. In this strategy, for each node in an $m \times n$ mesh a label $L(x, y)$ is assigned, where x and y are node's coordinates, as follows: $L(x, y) = y \times n + x$, if y is even, and $L(x, y) = y \times n + n - x - 1$, if y is odd. As shown in Fig. 2, two directed Hamiltonian paths (or two subnetworks) are constructed by the labeling. The up channel subnetwork (H_U) starts at (0, 0), and the down channel subnetwork (H_D) ends at (0, 0). In case the label of the destination node is greater than the label of the source node, the routing always takes place in the H_U subnetwork; otherwise it takes place in the H_D subnetwork. The destinations are placed into two groups. One group contains all the destinations that could be reached using the H_U subnetwork, and the other contains the remaining destinations that could be reached using the H_D subnetwork. To reduce the path length, the vertical channels that are not part of the Hamiltonian path (the dashed lines in Fig. 2) could be used in appropriate directions. The proposed adaptive method designed for both unicast and multicast messages, uses the Hamiltonian path strategy.

4.1. Multi-Path multicast routing

In the Multi-Path (MP) routing algorithm at first the destination node set is partitioned into two subsets, D_U and D_D, where every node in D_U has a higher label than the source node and every node in D_D has a lower label than the source node. In order to reduce the path lengths, D_U and D_D are also partitioned. The set D_U is divided into two subsets (D_U: D_{U1}, D_{U2}). One consist of the nodes whose x coordinates are greater than or equal to that of the source and the other subset contains the remaining nodes in D_U. The set D_D is partitioned in a similar way into two subsets ($D_{D:}$ D_{D1}, D_{D2}). Hence, all destinations of multicast message are grouped into four disjointed subnetworks. Consider the example in 8 × 8 mesh network, illustrated in Fig. 3(a) where source node 27 (3, 4) generates

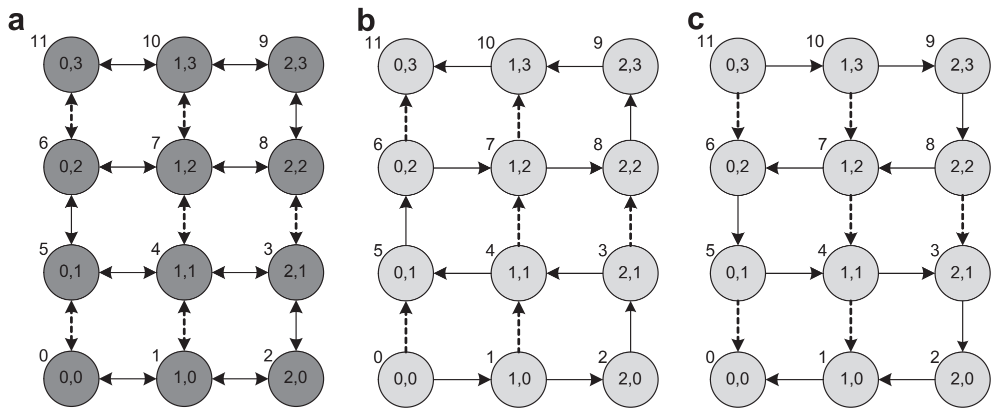

Fig. 2. (a) 3 × 4 mesh physical network with the label assignment and the corresponding (b) up channel and (c) down channel subnetworks. The solid lines indicate the Hamiltonian path and dashed lines indicate the links that could be used to reduce path length.

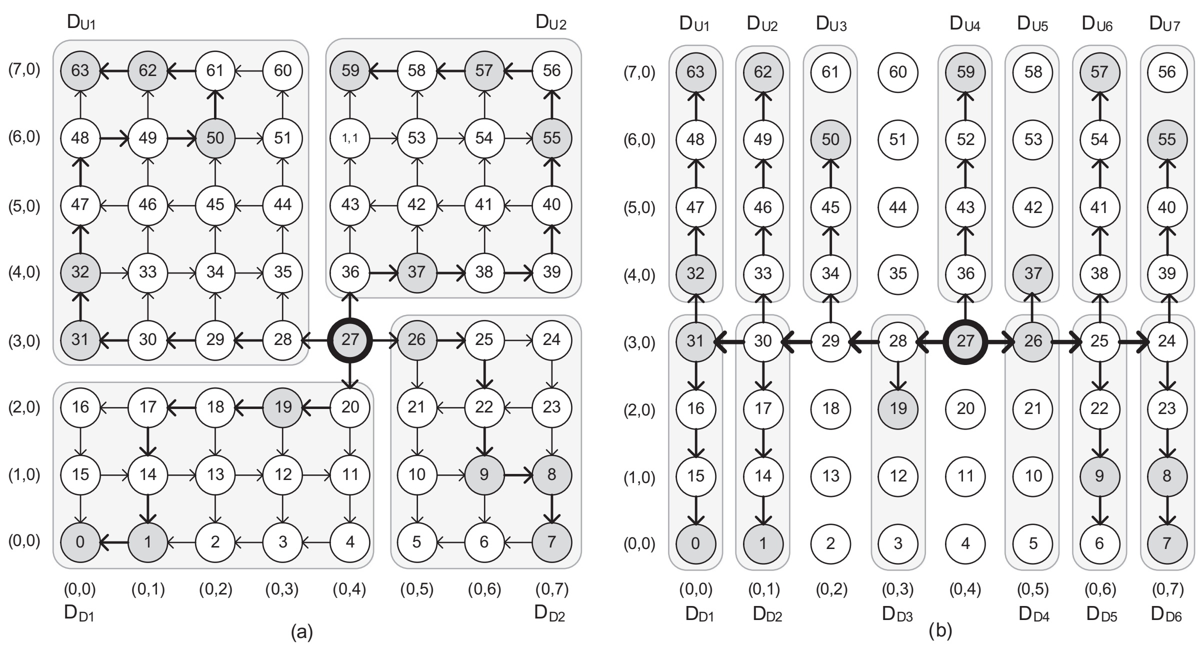

Fig. 3. (a) Multi-Path routing algorithm and (b) Column-Path routing algorithm.

a multicast message to be sent towards destinations 31, 9, 59, 8, 50, 57, 26, 19, 62, 37, 0, 63, 1, 7, 32, 55. Accordingly, two subsets are organized. The first subset (D_U) has all the destinations with higher label than the source node which are 31, 59, 50, 57, 62, 37, 63, 32, 55 and the second one (D_D) has the remaining destinations which are 9, 8, 26, 0, 1, 19, 7. As exhibited in Fig. 3(a), D_U is divided into two subsets, which are $D_{U1} = \{31, 50, 62, 63, 32\}$ and $D_{U2} = \{59, 57, 37, 55\}$. In the same way D_D is divided into two subsets, $D_{D1} = \{19, 0, 1\}$ and $D_{D2} = \{9, 8, 26, 7\}$. Subsequently, all destinations in D_{U1}, D_{U2} should be sorted in ascending order, $D_{U1} = \{31, 32, 50, 62, 63\}$, $D_{U2} = \{37, 55, 57, 59\}$, and the destinations in D_{D1}, D_{D2} should be sorted in descending order, $D_{D1} = \{19, 1, 0\}$ and $D_{D2} = \{26, 9, 8, 7\}$. Finally, one packet per subset should be created and sent from the source node to the network. All packets must follow the Hamiltonian path and reach to destinations in the order they are arranged. The Multi-Path is deterministic and deadlock-free algorithm that could be used for unicast and multicast routing simultaneously.

4.2. Column-Path multicast routing

In Column-Path (CP) algorithm, the destination node set is partitioned to $2k$ subsets. k is the number of columns in the mesh, and at most two messages will be copied to each column. If a column of the mesh has one or more destinations in rows above the source, then one copy of the message is sent to service all of those destinations. Similarly, if a column has one or more destinations in the rows below the source, then another copy of the message is sent to service all of those destinations. One copy of the message is sent to a column if all destinations in that column are either below or above the source node. Otherwise, two messages are sent to that column. An example is shown in Fig. 3(b) where a multicast message is generated to be sent towards destinations 31, 9, 59, 8, 50, 57, 26, 19, 62, 37, 0, 63, 1, 7, 32, 55 from the source node 27. Thirteen copies of the message are used to achieve the desired multicast operation. Though destinations 1 and 62 are in the same column, two message copies are sent to this column, since two of the destinations are above the source node's row and the other below. The routing algorithm used in this scheme is based on the XY routing algorithm. Therefore, the CP routing algorithm is compatible with the unicast routing method and it is deterministic, deadlock-free and livelock-free [13].

5. Hamiltonian Adaptive Multicast Unicast Method (HAMUM)

The traditional path-based routing models such as MP and CP, route the unicast and multicast messages by using deterministic routing algorithms. Therefore, the network performance is degraded by applying these algorithms. HAMUM can take the place of the deterministic model in the path-based routing algorithms to route both of the unicast and multicast messages through the destination(s). Fig. 4 shows the pseudo code of the HAMUM model which is executed in each router when a new packet arrives.

In this method, the locations where certain directions can be taken are restricted, so deadlock will be avoided. The rules regulating the proposed scheme are categorized in the up channel subnetwork and down channel subnetwork as follows:

For the up channel subnetwork:

Rule 1: North and East directions are allowed in even rows.
Rule 2: North and West directions are allowed in odd rows.

For the down channel subnetwork:

Rule 1: South and West directions are allowed in even rows.
Rule 2: South and East directions are allowed in odd rows.

Notice that a message will be forwarded to the destination as in the deterministic Hamiltonian strategy, when the current node is located one row to the south (north) of the destination row in the up channel subnetwork (down channel subnetwork). Inasmuch as the rules keep the messages traveling through the Hamiltonian paths, it prevents the occurrence of deadlock. In addition, both minimal and non-minimal paths are possible with HAMUM. However, our implementation is based on minimal paths and does not support the non-minimal paths.

5.1. Unicast aspect of HAMUM

Based on the proposed method, any intermediate node must first determine set of directions toward which a packet may be forwarded for the next hop based on Rule 1 and Rule 2. As mentioned previously, according to the source and destination labels, the routing may take place in up or down channel subnetwork. Consider a case where the destination of a message is to the west of its source in the up channel subnetwork (e.g. source node 7 and destination 27 in Fig. 5(b)). If the current node is in an odd row, the router can route the message to the west or north direction because of the Hamiltonian up channel subnetwork network strategy. If the current node is in an even row, at first the message

```
Algorithm HAMUM  is
-- (Cx,Cy) : Current  node , (Dx,Dy) : Destination node
Begin
    If (Dy = Cy) then                                     --Current & Dest. are in the same row
        If (Dx = Cx) then                                     --Current& Dest. are in the same column
             direction <= Local;                              --Packet sends to the Local direction
        Elsif (Dx > Cx) then                                  --Dest. is to the East of the Current
             direction <= East;                               --Dest. is to the West of the Current
        Else   direction <= West;
        End if;
    Elsif (Dy > Cy) then                                  -- up channel Subnetwork
        If ( Cy mod 2 = 0 ) then                              --rule1 in the even rows
            If ( Dx > Cx ) and ( Dy - Cy > 1 ) then           --Dest. is in the East & more than 1 row to Current
                 direction <= North or East                   --North or East direction can be chosen
            Elsif ( Dx > Cx ) and ( Dy - Cy = 1 ) then        --Dest. is in the East & 1 row to the Current
                 direction <= East;                           --Packet sends to the East direction
            Else   direction <= North;                        --IF Dest. is in the West of the Current, select North
            End if;
        Elsif ( Cy mod 2 /= 0 ) then                          --rule2 in odd rows
            If ( Dx < Cx ) and ( Dy - Cy > 1 ) then           --Dest. is in the West & more than 1 row to Current
                 direction <= North or West                   --North or West direction can be chosen
            Elsif ( Dx < Cx ) and ( Dy - Cy = 1 ) then        --Dest. is in the West & 1 row to the Current
                 direction <= West;                           --Packet sends to the West direction
            Else   direction <= North;                        --IF Dest. is in the West of the Current, select North
            End if;
        End if;
    Elsif ( Dy < Cy ) then                                -- down channel Subnetwork
        If ( Cy mod 2 = 0 ) then                              --rule1 in even rows
            If ( Dx < Cx ) and ( Cy - Dy > 1 ) then           --Dest. is in the West & more than 1 row to Current
                 direction <= South or West                   --South or West direction can be chosen
            Elsif ( Dx < Cx ) and ( Cy - Dy = 1 ) then        --Dest. is in the West & 1 row to the Current
                 direction <= West;                           --Packet sends to the West direction
            Else   direction <= South;                        --IF Dest. is in the West of the Current, select South
            End If;
        Elsif ( Cy mod 2 /= 0 ) then                          --rule2 in odd rows
            If ( Dx > Cx ) and ( Cy - Dy > 1 ) then           --Dest. is in the East & more than 1 row to Current
                 direction <= South or East                   --South or East direction can be chosen
            Elsif ( Dx > Cx ) and ( Cy - Dy = 1 )then         --Dest. is in the East & 1 row to the Current
                direction <= East;                            --Packet sends to the East direction
            Else  direction <= South;                         --IF Dest. is in the West of the Current, select South
            End if;
        End if;
    End if;
End HAMUM;
```

Fig. 4. The pseudo VHDL code of HAMUM.

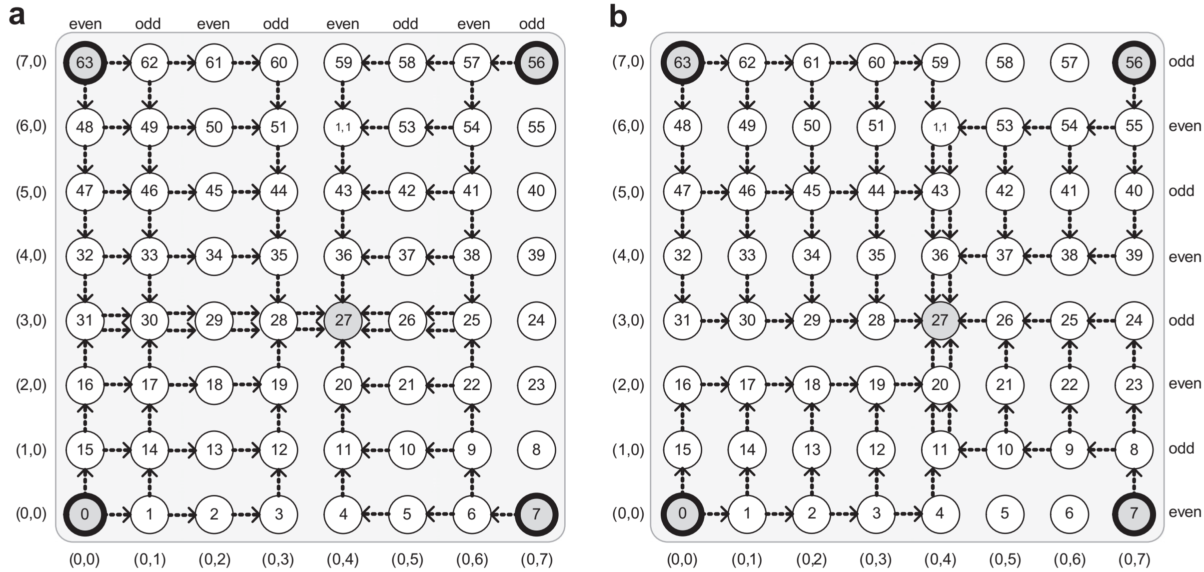

Fig. 5. All of the possible minimal paths from the source nodes 63, 56, 7, and 0 to the destination node 27 in (a) the Odd–Even model and (b) the unicast aspect of HAMUM.

should be routed to the north direction (to reach the odd row), and then, it could be routed via the west or north direction. Note that in the up channel subnetwork, using the Hamiltonian path, the packet can choose west or north direction in odd rows and east or north direction in even rows.

Additionally, if the current node is located one row to the south of the destination row in the up channel subnetwork, the message will be routed to the west or north direction if the current node is in the odd row, and if the current node is in the even row the packet will be routed to the north direction. In Fig. 5(b), all the possible minimal routing paths of HAMUM for four messages in 8 × 8 2D-mesh have been shown. At least one minimal path always can be selected by the proposed method for any source and destination pair. Since the Odd–Even model [20] is one of the most popular wormhole-based adaptive unicast routing algorithms in on-chip interconnection network, we compare the unicast aspect of our method with the Odd–Even model. All of the possible routing paths for the Odd–Even model are indicated in Fig. 5(a).

In order to compare the two algorithms with each other, we use the Degree of Adaptiveness (*DoA*) factor [23], which is the number of minimal paths can be taken by a message to travel from a source node (S_x, S_y) to a destination node (D_x, D_y). Suppose that Δx, Δy are defined as $\Delta x = D_x - S_x$ and $\Delta y = D_y - S_y$, and $d_x = |\Delta x|$ and $d_y = |\Delta y|$. The degree of adaptiveness for a fully adaptive algorithm is given by:

$$DoA(\text{fully adaptive routing})_{s,d} = \frac{(d_x + d_y)!}{d_x! d_y!}$$

Based on the Hamiltonian path, there can be eight different location states according to the source node position (even or odd row), destination node position (even or odd row), and the direction of the destination node (left or right side of the source node). The states have been summarized in Table 1.

First, we compute the *DoA* for unicast messages in the up channel subnetwork, then we use the similar way to compute the *DoA* for the down channel subnetwork. As can be seen in Fig. 6, the *DoA* of the state 1 and 8 is equal and can be computed as:

$$DoA(1)_{s,d} = \frac{(d_x + D)!}{d_x! D!}, \quad \text{where } D = \left\lfloor \left| \frac{d_y}{2} \right| \right\rfloor$$

For the other states, the *DoA* function is calculated as:

Table 1
Eight different location states of the source and destination nodes.

State	Source position (odd/even row)	Destination position (odd/even row)	Destination direction (left/right)
1	Even	Even	Right (east)
2	Even	Odd	Right (east)
3	Even	Even	Left (west)
4	Even	Odd	Left (west)
5	Odd	Even	Right (east)
6	Odd	Odd	Right (east)
7	Odd	Even	Left (west)
8	Odd	Odd	Left (west)

$$DoA(2)_{s,d} = \frac{(d_x + D')!}{d_x! D'!}, \quad \text{where } D' = \left\lfloor \left| \frac{d_y - 1}{2} \right| \right\rfloor$$

These equations can be summarized as:
DoA of the up channel subnetwork:

$$DoA(\text{up-channel})_{s,d} = \begin{cases} DoA(1)_{s,d} & \text{for conditions 1 and 8} \\ DoA(2)_{s,d} & \text{otherwise} \end{cases}$$

DoA of the down channel subnetwork:

$$DoA(\text{down-channel})_{s,d} = \begin{cases} DoA(1)_{s,d} & \text{for conditions 3 and 6} \\ DoA(2)_{s,d} & \text{otherwise} \end{cases}$$

The Odd–Even [20] model restricts the locations where some types of turns can be taken. While HAMUM rules are based on the mesh rows, the rules of the Odd–Even model are based on the columns. Odd–Even rules are described as follows:

Rule 1: East-North and East-South turns cannot be taken in even columns (Fig. 7(a)).
Rule 2: North-West and South-West turns cannot be taken in odd columns (Fig. 7(b)).

The degree of adaptiveness for the Odd–Even turn model is computed as [20]:

When the destination node is in the right side of the source node ($\Delta x > 0$):

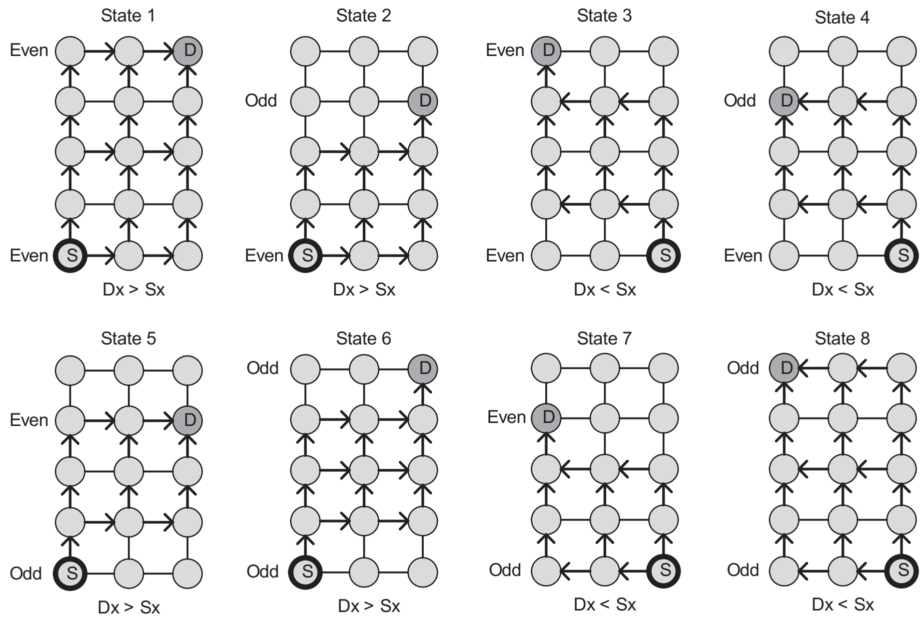

Fig. 6. Eight different location states in the up channel subnetwork.

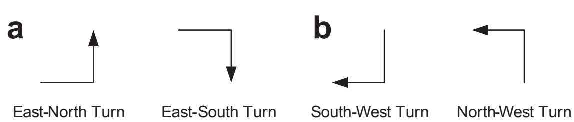

Fig. 7. The Odd–Even turn model rules: (a) prohibited turns in even columns and (b) prohibited turns in odd columns.

$$DoA(\Delta x > 0)_{s,d} = \begin{cases} DoA(2)_{s,d} & \text{if source node's column is an allowable column, destination is in odd column} \\ DoA(1)_{s,d} & \text{otherwise} \end{cases}$$

When the destination node is to the left side of the source node ($\Delta x < 0$):

$$DoA(\Delta x < 0)_{s,d} = \begin{cases} DoA(1)_{s,d} & \text{if source node's column is an allowable column, and } \Delta x = 0 \\ DoA(2)_{s,d} & \text{otherwise} \end{cases}$$

Considering the above analysis, the degree of adaptiveness of HAMUM and the Odd–Even models is equal. Since the Odd–Even model cannot be utilized for the multicast traffic, described in the motivation, HAMUM not only is compatible with multicast traffic but also provides adaptivity for both unicast and multicast traffic.

5.2. Multicast aspect of HAMUM

In this section, we describe how the proposed adaptive method affects the path-based multicast routing algorithms. For this purpose, we apply HAMUM on the Multi-Path (MP) and Column-Path (CP) algorithms.

5.2.1. AMP routing algorithm

AMP, Adaptive MP, is the adaptive model of the MP algorithm after the proposed adaptive model is applied in the MP algorithm. Consider the example used for MP in Fig. 8(a). The multicast message can be forwarded in three different ways from the node 37 through the node 55 (32 through 50, 19 through 1, and 26 through 8).

5.2.2. ACP routing algorithm

The ACP, stood for the adaptive CP is the adaptive method of the original CP by taking advantage of the proposed adaptive model. To indicate how the adaptive scheme affects the CP algorithm, as illustrated in Fig. 8(b), again 13 copies of the multicast message must be used to achieve the desired multicast operation. But in this figure for simplicity, we only consider two subsets D_{U2} and D_{D6}. Due to utilizing the proposed adaptive scheme in the CP, each multicast messages can be delivered to its subset through different paths indicated by dashed lines.

6. Hardware Implementation

In this work, due to scalability, cross-section bandwidth, we use an $n \times n$ network of interconnected tiles with a mesh topology [14,24]. Each tile is composed of a PE (Processing Element) and a router connected to its four adjacent routers in addition to the PE of the tile through some channels. Two unidirectional point-to-point links form the channel. To minimize the delay and the required resources, we have used the wormhole method for the switching. In this method, a message is divided into smaller segments called FLITs (FLow control digIT) which are routed successively until they reach their destination [18].

6.1. Message format

The multicast message format is shown in Fig. 9. It includes one or several header flits and a parametric number of payload flits. The number of header flits depends on the number of destinations and the flit width in a multicast packet. Each flit is n bit wide and the nth bit is the EOM (End Of Message) sign and the $(n-1)$th bit is the BOM (Begin Of Message) sign. In the header, the third field is used to describe the type of the message. There are two types of messages: unicast ($T = 0$) and multicast ($T = 1$), indicated by T. The specific address of the source node, the pointer counter, and the destination node address(es) are placed in the last field of the header, respectively, and the content of the message is located

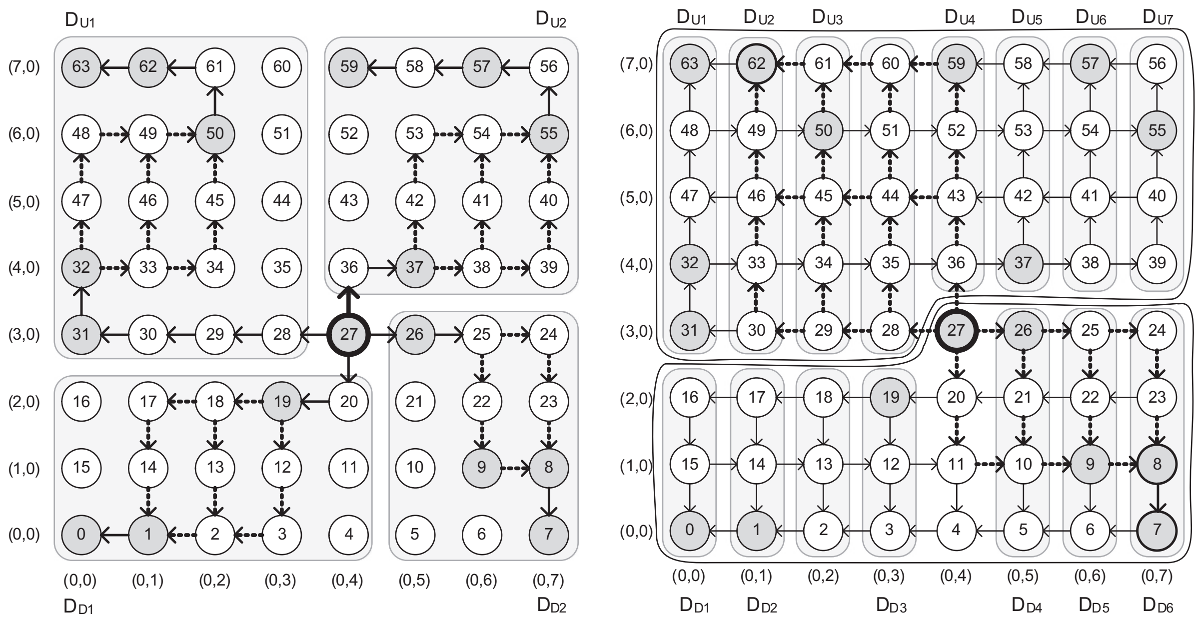

Fig. 8. (a) Adaptive Multi-Path (AMP) routing algorithm and (b) Adaptive Column-Path (ACP) routing algorithm.

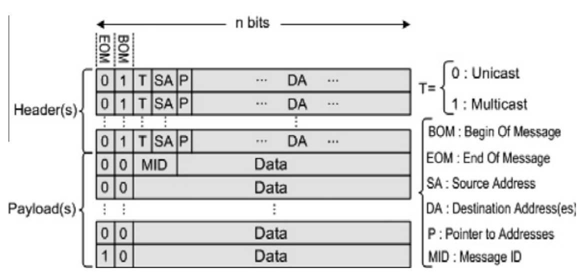

Fig. 9. Multicast message format [11].

in the rest of the flits (Payload). The pointer indicates the address of the next destination in the header flit, and the MID used for message ordering.

6.2. Router structure

As shown in Fig. 10 each input port has a controller for handshaking and an input buffer used for the temporary storage of flits. The wormhole switching method implemented in the controller unit, is based on on/off flow control mechanism [25]. After receiving the message header, the routing unit determines which output should be used for routing this message and then the arbiter requests for a grant to inject the message to a proper output using a crossbar switch. The router has the crossbar to create a path from an input port to an output port. Since the crossbar can only serve a single output port at a time, it arbitrates among simultaneous input requests to access the same output port. When a new message reaches the input port, it waits until the previously arrived messages leave the port. Then the header of the new message is delivered to the routing unit and routed to the appropriate output port. The Congestion Flag (CF) of the buffer becomes active when the number of empty cells of the buffer is less than a threshold value. In this case, warning for the full status, the signal CF, is activated indicating that most buffer cells are occupied. Each input port has a CF through which it informs its adjacent routers about its congestion condition. Therefore, the router which uses that input port for forwarding a message to the next router should consider this router as a congested one (congestion area or hotspot) and should not send messages to this router until the congestion condition is over.

6.3. Consumption channel deadlock

In the path-based multicast wormhole mechanism, when several delivery channels are occupied by one message along the multicast path, cyclic dependencies on the delivery channels may occur [13,16,17]. To prevent deadlocks in delivery (consumption) channels, the upper bound of the number of delivery channels required to avoid such deadlocks is equal to $2nv$, where n is the network dimension and v is the number of virtual channels per input port [13,16,17]. As a result, at least two delivery channels are necessary and sufficient for MP, AMP, CP and ACP algorithms [16,17].

6.4. Header processing mechanism

The router employs a routing unit which decodes the header of messages coming from an input port. If the header belongs to a unicast message ($T = 0$), the minimal path adaptive routing algorithm is used to determine the output port to which the message should be sent. In the proposed adaptive routing algorithm there could be more than one minimal output directions where to route messages. In this case the address decoder will choose the direction where the corresponding downstream router has not raised its Congestion Flag. For instance, if a message with a given source and destination could be routed to both output p1 (CF = 0) and p2 (CF = 1), then it will be routed to p1. If p1 and p2 happen to have both their Congestion Flag raised or fallen, the message will be routed to p1. On the other hand, if the header type is a multicast message ($T = 1$), the routing unit fetches the destination address from where the pointer in the header points. Afterward, the routing unit increases the pointer value of the header, and if it overflowed, the routing unit would remove the corresponding flit header from the message. In sum, whenever a destination address is fetched from the header, the pointer value will be increased. After fetching the destination address from the header, if the destination address is the current node, the routing unit will request the local output port. Meanwhile, the routing unit fetches the next destination ad-

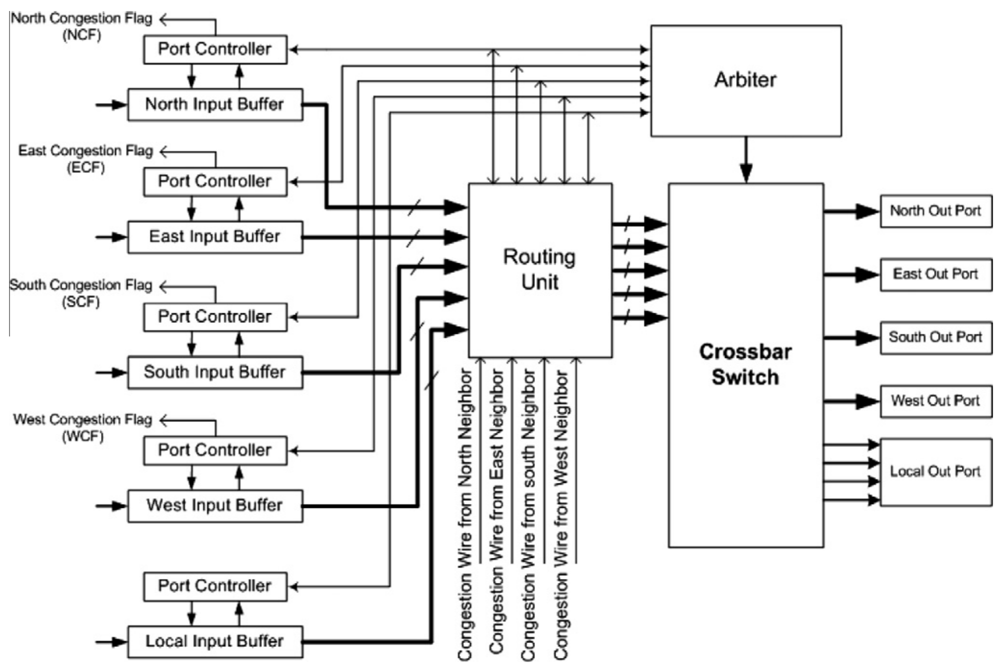

Fig. 10. The proposed router structure [11].

dress from the header and runs the adaptive routing procedure to determine the output port(s) corresponding to the next destination address.

7. Results and discussion

To assess the efficiency of the proposed adaptive method, two multicast routing algorithms were implemented. These algorithms include MP and CP. We have developed a synthesizable wormhole NoC simulator implemented in VHDL to assess the efficiency of the proposed adaptive method. This simulator can be used for wormhole switching in two-dimensional mesh configuration. The simulator inputs include the array size, the routing algorithm, the link width, buffer size, and the traffic type. The simulator can generate different traffic profiles. To calculate the power consumption, we have used power compiler. For all routers, the data width was set to 32 bits, and each input channel has a buffer (FIFO) size of 12 flits with the congestion threshold set at 75% of the total buffer capacity. The message size was assumed to be 16 flits. For the performance metric, we use the multicast latency defined as the number of cycles between the initiation of multicast message operation and the time when the tail of the multicast message reaches all the destinations.

7.1. Performance evaluation

7.1.1. Multicast traffic profile

The first set of simulations was performed for a random traffic profile. Two array sizes have been considered 8×8 and 16×16. In the multicast traffic profile, each PE sends a message to a set of destinations. A uniform distribution is used to construct the destination set of each multicast message [6]. The number of destinations has been set to 10 and 25. The average communication delay as a function of the average flit injection rate has been shown in Figs. 11 and 12. As observed from the results, the proposed adaptive mechanism which has been applied to MP and CP even in high traffic loads or with a large number of destinations (25 destinations) leads to lower delay.

7.1.2. Unicast and multicast (mixed) traffic profile

In this set of simulations, we have employed a mixture of unicast and multicast traffic, where 80% of injected messages are unicast messages and the remaining 20% are multicast messages. This pattern may be representative of the traffic in a distributed shared-memory multiprocessor where updates and invalidation produce multicast messages and cache misses are served by unicast messages [13,15,16]. The unicast messages are also routed using the proposed adaptive scheme. Uniform [20] and hotspot [20] are

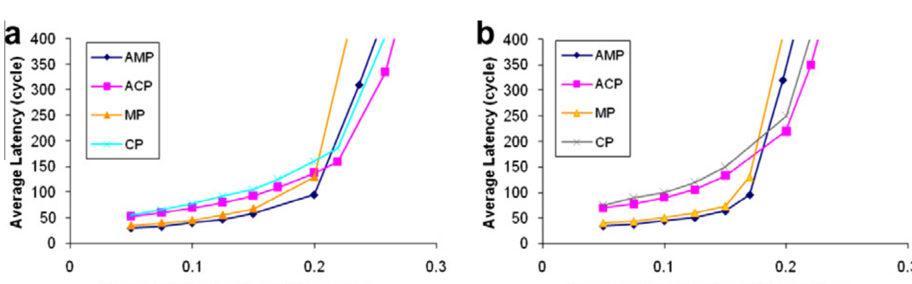

Fig. 11. Performance under different loads in 8×8 2D-mesh with (a) 10 destinations and (b) 25 destinations.

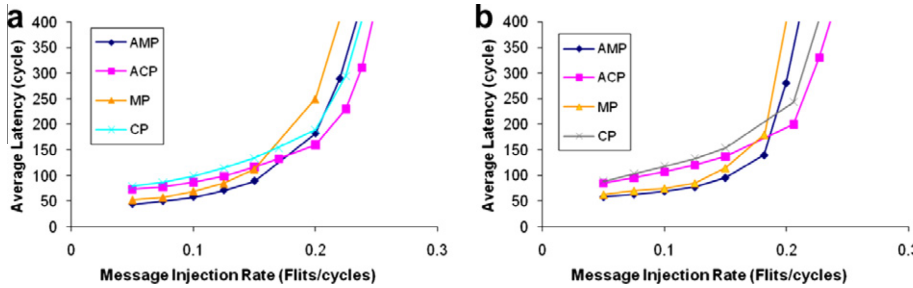

Fig. 12. Performance under different loads in 16 × 16 2D-mesh with (a) 10 destinations and (b) 25 destinations.

two different traffic profiles that have been taken into account for unicast traffic generation. In the uniform traffic profile, each PE sends a message to any other PE in an equal probability. This is determined randomly using a uniform distribution. Under the hotspot traffic pattern, one or more nodes are chosen as hotspots receiving an extra portion of the traffic in addition to the regular uniform traffic.

In Fig. 13 the average communication latency of different algorithms under the uniform traffic model for unicast traffic is shown. As depicted in these figures, for this traffic, the adaptive routing algorithms perform better. Under the hotspot traffic model, given a hotspot percentage of h, a newly generated message is directed to each hotspot node with an additional h percent probability. We simulate hotspot traffic with a single hotspot node. The hotspot node is chosen to be node (4, 4) in the 8 × 8 2D-mesh. Fig. 14 shows the multicast routing performance with h = 10%. As the figure shows, the adaptive proposed routing algorithm outperforms the traditional algorithms.

7.1.3. Application traffic profile

In order to know the real impact of the proposed model, we used traces from some application benchmark suites selected from SPLASH-2 [26] and PARSEC [27,28]. Traces are generated from SPLASH and PARSEC using the GEMS simulator [29]. We used the ×264 application of PARSEC and the Radix, Ocean, and fft applications from SPALSH-2 for our simulation. Table 2 summarizes our full system configuration. It is noteworthy that the token-based MOESI protocol [30] is heavily based on multicast. On account of our analysis on average 95% of token-based MOESI traffic is multicast.

As can be seen from Fig. 15, the proposed adaptive model diminishes the average delay of MP and CP significantly under all benchmarks. That is, adaptive routing has an opportunity to improve performance. Under the fft application the adaptive model indicates 17% and 21% reduction in latency for MP and CP, respectively.

7.2. Hardware overhead

To evaluate the area overhead of the proposed method, the routers were synthesized with Synopsys DC using the TSMC 0.09 μm standard cell library. In addition, the destination sorting algorithms are included in the hardware overhead. For all routers, the data width was set to 32 bits (flit size), and each input channel had a buffer size of 12 flits. As discussed earlier, for the MP, and CP routers we use two delivery channels. In order to

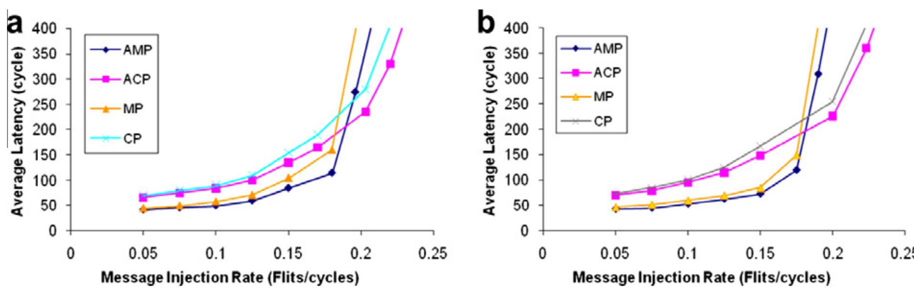

Fig. 13. Performance under different loads in 8 × 8 2D-mesh with (a) 10 destinations and (b) 25 destinations under mixed traffic (20% multicast and 80% unicast). Unicast traffic is based on the uniform traffic model.

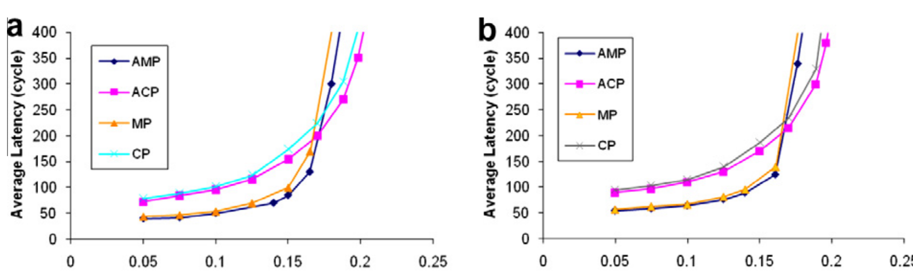

Fig. 14. Performance under different loads in 8 × 8 2D-mesh with (a) 10 destinations and (b) 25 destinations under mixed traffic (20% multicast and 80% unicast). Unicast traffic is based on the hotspot traffic model with a single hotspot node (4, 4), and h = 10%.

Table 2
System configuration parameters.

Processor configuration	
Instruction set architecture	SPARC
Number of processors	16
Issue width	1
Cache configuration	
L1 cache	Private, split instruction and data cache, each cache is 16 KB. 4-way associative, 64-bit line, 3-cycle access time
L2 cache	Shared, distributed in 3 layers, unified 48 MB (48 banks, each 1 MB). 64-bit line, 6-cycle access time
Cache coherence protocol	Token-based MOESI
Cache hierarchy	SNUCA
Memory configuration	
Size	4 GB DRAM
Access latency	260 cycles
Requests per processor	16 outstanding
Network configuration	
Router scheme	Wormhole
Flit size	32 bits

achieve better performance/power efficiency, the FIFOs were implemented using registers. The CP and MP multicasting schemes used the same router structure for the implementation, but their sorting mechanisms uses different number of registers. Comparing the area cost indicates that the hardware overhead of implementing the proposed adaptive scheme in both the MP and CP routers is less than 0.5% and that can be considered negligible.

7.3. Power dissipation

The power dissipation of MP, CP, AMP, and ACP were calculated and compared under the multicast traffic model with 25 destinations using Synopsys PrimePower. The typical clock of 1 GHz is applied to the 8×8 2D-mesh network. The results for the average and maximum power under this traffic are shown in Fig. 16(a) and (b), respectively. As the results presented in Table 3 reveal,

Table 3
Comparative average power dissipation of the proposed algorithms with other algorithms in 8×8 2D-mesh.

Average power dissipation	AMP/MP	ACP/CP
With 25 destinations	3.5%	5%

Table 4
Comparative maximum power dissipation of the proposed algorithms with other algorithms in 8×8 2D-mesh.

Maximum power dissipation	AMP/MP	ACP/CP
With 25 destinations	11%	15%

the average power dissipation of the network with the ACP algorithm is 5% less than that of the CP algorithm and the average power dissipation of the AMP is 3.5% less than that of the MP algorithm. The results of Table 4 indicate that the peak power of the ACP and AMP algorithms is 15% and 11% less than that of the CP and MP algorithms, respectively, under the multicast traffic model. We can notice that the average power and the peak power of the proposed adaptive models are lower. This is achieved by smoothly distributing the power consumption over the network using the adaptive routing scheme which reduces the number of the hotspots and, hence, lowering both the average power and the peak power.

8. Summary and conclusion

In this paper, an adaptive method based on the Hamiltonian path in mesh interconnection networks for NoCs was proposed. In this scheme, three facets have been considered such as utilization of network partitioning, and taking advantage of the proposed adaptive method for routing both the multicast and unicast messages through the network. Additionally, the adaptive routing algorithm used the congestion condition of the input ports to route messages through non-congested paths while it enabled us to distribute the traffic load over the network. A synthesizable VHDL

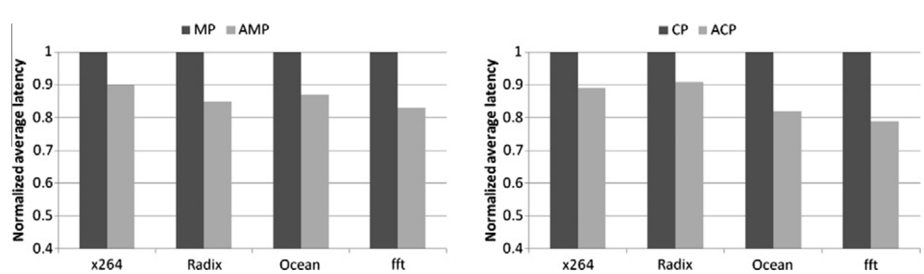

Fig. 15. Performance under different application benchmarks for Multi-Path (left) and Column-Path (right) routing algorithms.

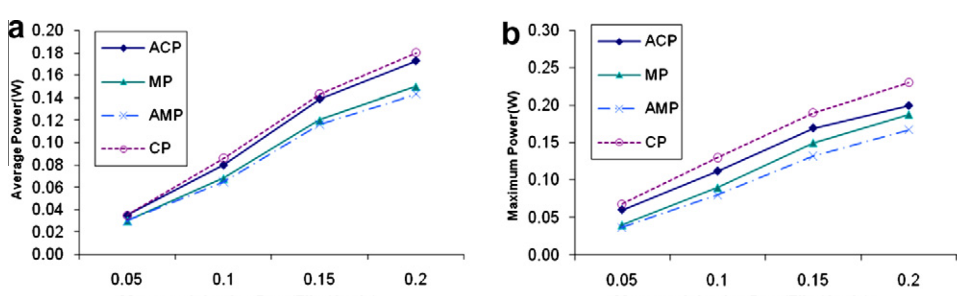

Fig. 16. (a) Average and (b) maximum power dissipation of the MP, CP, AMP, ACP algorithms in 8×8 2D-mesh with 25 destinations under multicast traffic.

NoC-environment was developed to evaluate the efficiency of the proposed multicast routing algorithm. Under the multicast, mixed (mixture of unicast and multicast), and real traffic models and in high flit injection rates, the proposed adaptive models had lower average communication delay in comparison with the traditional Multi-Path, and Column-Path multicast routing algorithms.

Acknowledgment

The authors wish to acknowledge the academy of Finland and Nokia Foundation for the financial support during the course of this research.

References

[1] A. Jantsch, H. Tenhunen, Networks on Chip, Kluwer Academic Publishers, 2003.
[2] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, S. Borkar, A 5-GHz mesh interconnect for a teraflops processor, IEEE Micro 27 (September–October) (2007) 51–61.
[3] W.O. Cesario, D. Lyonnard, Y. Paviot, S. Yoo, M. Gauthier, M. Diaz-Nava, A.A. Jerraya, Multiprocessor SoC platforms: a component-based design approach, in: Proceedings of the International Conference on IEEE Design and Test of Computers, Paris, France, 2002, pp. 52–63.
[4] E. Rijpkema, K.G.W. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, E. Waterlander, Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip, in: Proceedings of the International Conference on Design Automation and Test Conference in Europe, Munich, Germany, 2003, pp. 350–355.
[5] Z. Lu, B. Yin, A. Jantsch, Connection-oriented multicasting in wormhole-switched networks on chip, in: Proceedings of the International Conference on ISVLSI, Germany, 2006, pp. 205–210.
[6] X. Lin, L.M. Ni, Multicast communication in multicomputer networks, IEEE Transactions on Parallel and Distributed Systems (1993) 1105–1117.
[7] K. Li, R. Schaefer, A hypercube shared virtual memory, in: Proceedings of the International Conference on ICPP, USA, 1989, pp. 125–132.
[8] M. Azevedo, D. Blough, Fault-tolerant clock synchronization of large multicomputers via multistep interactive convergence, in: Proceedings of the International Conference on ICDCS, Hong Kong, 1996, pp. 249–257.
[9] P.K. McKinley, H. Xu, E. Kalns, L.M. Ni, CompaSS: efficient communication services for scalable architectures, in: Proceedings of the International Conference on supercomputing, Minnesota, USA, 1992, pp. 478–487.
[10] H. Xu, P.K. McKinley, E. Kalns, L.M. Ni, Efficient implementation of barrier synchronization in wormhole-routed hypercube multicomputers, Journal of Parallel and Distributed Computing 16 (1992) 172–184.
[11] M. Daneshtalab, M. Ebrahimi, S. Mohammadi, A. Afzali-Kusha, Low distance path-based multicast algorithm in NOCs, IET – Computers and Digital Techniques 3 (2009) 430-442 (Special issue on NoC).
[12] P. Mohapatra, V. Varavithya, A hardware multicast routing algorithm for two-dimensional meshes, in: Proceedings of the International Conference on SPDP, New Orleans, 1996, pp. 198–205.
[13] R.V. Boppana, S. Chalasani, C.S. Raghavendra, Resource deadlock and performance of wormhole multicast routing algorithms, IEEE Transactions on Parallel and Distributed Systems (1998) 535–549.
[14] N.E. Jerger, L.S. Peh, M.H. Lipasti, Virtual circuit tree multicasting: a case for on-chip hardware multicast support, in: Proceedings of the International Conference on Computer Architecture, China, 2008, pp. 229–240.
[15] P. McKinely, H. Xu, A.H. Esfahanian, L. Ni, Unicast-based multicast communication in wormhole-routed networks, IEEE Transactions of Parallel and Distributed Systems 5 (1994) 1252–1265.
[16] M. Malumbres, J. Duato, J. Torrellas, An efficient implementation of tree-based multicast routing for distributed shared-memory multiprocessors, in: Proceedings of the International Conference on SPDP, New Orlean, USA, 1996, pp. 186–190.
[17] E.A. Carara, F.G. Moraes, Deadlock-free multicast routing algorithm for wormhole-switched mesh networks-on-chip, in: Proceedings of the ISVLSI, 2008, pp. 341–346.
[18] J. Duato, S. Yalamanchili, L.M. Ni, Interconnection Networks: An Engineering Approach, Morgan Kaufmann Publishers, 2003.
[19] A. Al-Dubai, I. Romdhani, A performance study of path based multicast communication algorithms, in: Proceedings of the International Conference on PARELEC, Bialystok, Poland, 2006, pp. 245–250.
[20] G. Chiu, The Odd–Even turn model for adaptive routing, IEEE Transactions on Parallel and Distributed System (July) (2000) 729–738.
[21] J.C. Hu, R. Marculescu, DyAD – smart routing for networks-on-chip, in: Proceedings of the Design Automation Conference, 2004, pp. 260–263.
[22] F. Harary, Graph Theory, Addison-Wesley, Readings, Massachusetts, 1972.
[23] C.J. Glass, L.. Ni, The turn model for adaptive routing, in: Proceedings of the 19th Annual International Symposium on Computer Architecture, May 1992, pp. 278–287.
[24] J. Liang, S. Swaminathan, R. Tessier, aSOC: a scalable, single-chip communication architectures, in: Proceedings of the International Conference on PACT, Oregon, USA, 2000, pp. 37–46.
[25] N. Concer, M. Petracca, L. Carloni, Distributed flit-buffer flow control for networks-on-chip, in: Proceedings of the International Conference on CODES + ISSS, Georgia, USA, 2008, pp. 215–220.
[26] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta, The splash-2 programs: characterization and methodological considerations, in: Proceedings of the 22nd International Symposium on Computer Architecture (ISCA), 1995, pp. 24–36.
[27] C. Bienia, S. Kumar, J.P. Singh, K. Li, The parsec benchmark suite: characterization and architectural implications, in: Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, 2008, pp. 72–81.
[28] C. Bienia, S. Kumar, K. Li, Parsec vs. splash-2: a quantitative comparison of two multithreaded benchmark suites on chipmultiprocessors, in: IEEE International Symposium on Workload Characterization, 2008, pp. 47–56.
[29] M.M.K. Martin et al., Multifacet's general executiondriven multiprocessor simulator (GEMS) toolset, SIGARCH Computer Architecture News 33 (4) (2005) 92–99.
[30] M. Martin, M. Hill, D. Wood, Token coherence: decoupling performance and correctness, in: Proceedings of the 30th Annual International Symposium on Computer Architecture (ISCA), 2003, pp. 182–193.

Masoud Daneshtalab received his Master's degree in computer architecture from School of Electrical and Computer Engineering, University of Tehran in 2006. Since autumn 2008 he has been working in the Computer Systems Laboratory, University of Turku and from May 2009 he is a doctoral candidate of Graduate School in Electronics, Telecommunications and Automation (GETA). He is expected to get his Ph.D. degree in January 2011. He has expertise in on/off-chip interconnection networks, multiprocessor architectures, Network-on-Chips (NoC), and low-power digital design. His Ph.D. thesis is focused on topology formation and routing protocol in 2D and 3D On-chip Networks.

Masoumeh Ebrahimi received his Master's degree in computer architecture from Azad University, Science and research branch, in 2009. Since spring 2009 she has been working in the Computer Systems Laboratory, University of Tutku. She has expertise in on/off-chip interconnection networks, multiprocessor architectures, Network-on-Chips (NoC), and low-power digital design. Her Ph.D. thesis is focused on routing protocols in 2D and 3D NoCs.

Thomas Canhao Xu received his M.Eng. degree in Software Engineering from Zhejiang University, China in 2007. He has been teaching the National Certification of Information Engineer (NCIE) and Wish certified Network Engineer (WNE) for two and half years. Since September 2008, he has been working in the Computer Systems Laboratory, University of Turku as a researcher. He is also a Ph.D. student in the Turku Centre for Computer Science (TUCS), Turku, Finland. His research interests include software system support for Network-on-Chip platforms, system level 3D multiprocessor architecture design and software engineering.

Pasi Liljeberg received his Ph.D. degree in Electronics and Information Technology from the University of Turku in 2005. Since January 2010 he has been working in the Computer Systems laboratory, University of Turku as a senior lecturer. During the period 2007–2009 he has worked as an Academy of Finland postdoctoral researcher. He has supervised one Ph.D. thesis, one Lic.Tech. thesis and eight M.Sc. theses, and is currently supervising six Ph.D. students. His current research interests include Network-on-Chip intelligent communication architectures, on-chip fault tolerant design aspects, 3D multiprocessor system architectures, self-timed design and formal approaches in embedded system development.

Prof. Hannu Tenhunen received his PhD from Cornell University, Ithaca, USA in 1985 and since that he has held professor, invited professor, or honorary professor positions in Tampere, Stockholm, Ithaca, Grenoble, Shanghai, Beijing and Hong Kong. During the recent years he has been director of Turku Centre of Computer Science and invited professor at University of Turku where he has established Computer Systems Laboratory, the leading computer architecture and systems research centre in Finland. Prof. Tenhunen's research interest is in new computational architectures, dependability issues, on-chip and off-chip communication and mixed signal and interference issues in complex electronic systems including 3D integration.