Research Article

Bus Based Synchronization Method for CHIPPER Based NoC

D. Muralidharan and R. Muthaiah

SASTRA University, Thanjavur, Tamil Nadu 613 401, India

Correspondence should be addressed to D. Muralidharan; murali@core.sastra.edu

Received 28 January 2016; Revised 12 May 2016; Accepted 7 June 2016

Academic Editor: Meng Guo

Copyright © 2016 D. Muralidharan and R. Muthaiah. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Network on Chip (NoC) reduces the communication delay of System on Chip (SoC). The main limitation of NoC is power consumption and area overhead. Bufferless NoC reduces the area complexity and power consumption by eliminating buffers in the traditional routers. The bufferless NoC design should include live lock freeness since they use hot potato routing. This increases the complexity of bufferless NoC design. Among the available propositions to reduce this complexity, CHIPPER based bufferless NoC is considered as one of the best options. Live lock freeness is provided in CHIPPER through golden epoch and golden packet. All routers follow some synchronization method to identify a golden packet. Clock based method is intuitively followed for synchronization in CHIPPER based NoCs. It is shown in this work that the worst-case latency of packets is unbearably high when the above synchronization is followed. To alleviate this problem, broadcast bus NoC (BBus NoC) approach is proposed in this work. The proposed method decreases the worst-case latency of packets by increasing the golden epoch rate of CHIPPER.

1. Introduction

In System on Chip (SoC), many processors are added in the same chip to enhance processing speed. Frequently they share information among themselves. This communication is done through bus medium. Since only one processor can use the bus at a time, the communication speed is reduced notably. Network on Chip (NoC) reduces this communication bottleneck. The concepts and architectures for NoC are discussed in studies [1–4]. NoC design methodologies are discussed in [5]. In NoC, every processor is connected to its router and communication among processors is done through a network of routers. A router is connected to four neighbor routers which are placed in four cardinal directions. Additionally it is connected to its host processor. Hence the routers in NoC are called five port routers. The routers can be connected in various interconnection methods, which lead to different topologies as given in [6, 7]. Mesh and torus are two popular NoC topologies. Figures 1(a) and 1(b) show that routers connected in 4×4 mesh topology and 4×4 torus topology, respectively. As shown in Figures 1(a) and 1(b), the main difference between these two topologies is the connection among end routers. The additional connections

reduce the packet latency but increase the area in torus topology compared to mesh topology. In torus all routers have five ports. So, torus is a regular topology. In mesh, a router can have three, four, or five ports depending on its position in the network.

When two or more packets from different input ports are competing for the same output port, only one packet is considered as winner and is assigned to that port. The other packets are considered as losers. The losers have to be either discarded or saved in the router till the port is free for them. Second option is considered in NoC to enhance the performance. Hence routers of NoC have buffers. Additionally, these buffers help to increase simultaneous transmissions with the cost of reduced bandwidth. This concept was suggested by Lu and Jantsch [8] as virtual channel mechanism.

But buffers have limitations. Buffers cause increment of power consumption in routers as discussed in a study [9]. Obviously the area overhead is also increased. Various research proposals are presented to mitigate these problems. Nicopoulos et al. proposed area and power reduction by using dynamic buffer space in the form of virtual channel regulator [10]. Power reduction methods are given in [11, 12]. But the simplest solution is removing buffers from routers.

FIGURE 1: (a) 4×4 mesh NoC. (b) 4×4 torus NoC.

The obvious drawback in this technique is packet loss since there is no buffer to save the packets. When the intended port is not available, Moscibroda and Mutlu [13] suggest assigning these packets to any one of the free ports instead of saving in buffer. In this way packet loss is avoided. This NoC is called as Bufferless Network on Chip (BLESS NoC). This is based on hot potato routing which was first explained by Baran [14].

BLESS NoC improves power and area consumptions. Furthermore, the throughput reduction is not severe. In hot potato or deflection routing technique the desired output link is termed as productive link and others are called as nonproductive links. When there is a competition for a productive link among multiple packets, one is chosen as winner and is assigned to the productive port. Other packets are deflected to any available nonproductive ports. This deflection raises a problem called live lock. When a packet is live locked, it is moving towards and away from the destination but cannot reach it. Bufferless algorithms should be free from this problem. When the number of deflections is increased packet latency is also increased. Therefore, it is vital to decrease the number of deflections to avoid live lock and latency. Various methods to decrease the deflection count for both buffered and bufferless are suggested in [15–17]. The main limitation in these methods is the introduction of storage elements. Hence these are not bufferless NoC.

In BLESS the deflection count is incremented when a packet is deflected and the packet with highest number of deflection count is chosen as winner during competitions. Performance of BLESS NoC is comparable to buffered NoC. The main limitation of BLESS is the number of bits allocated for deflection count. This reduces the ratio of data bits in flow control unit (flit) or packet. To keep the same number of data bits in a packet, the wires which connect two routers have to be increased. This increases the area overhead. Otherwise more packets have to be sent to convey a message. This increases the traffic and increases the probability of deflection.

The overhead of deflection count is eliminated in CHIPPER [18]. In CHIPPER, a packet is decided as a golden packet

by all the routers of the network. For a predetermined time period this packet has the highest priority among all packets. This time period is called as golden epoch. Golden packet is selected by all routers of the network in a predetermined manner as given below.

Obviously all packets in the network have source id and packet id. Initially the packet with packet id “0” of source “0” is assumed as golden packet. In the next golden epoch, packet id “0” of source “1” is selected as golden packet. Concisely, in a current golden epoch, the packet number “ p ” from the source processor “ s ” is considered as golden packet; then in the next golden epoch, the packet number “ p ” from the source processor “ $s + 1$ ” is considered as golden packet. When “ s ” is the last processor in the network, then the packet number “ $p + 1$ ” from the first processor is assigned as the golden packet in the next golden epoch. In this way CHIPPER guarantees at least one packet delivery in one golden epoch. Eventually all packets will be delivered. So, it is a live lock free algorithm. Furthermore, it also eliminates the bits for deflection count in flit. As a result, the data ratio in a flit is increased. Due to these advantages CHIPPER is considered as a low overhead, live lock free, bufferless technique.

Throughput enhancement of CHIPPER is analyzed in MD [19] and MINBD [17]. In MD, minimum deflection is employed instead of random selection at the time of selecting an outport. In this way it increases the throughput. MINBD uses minimum buffer area in routers to reduce deflection. This helps to improve throughput. But it is not a pure bufferless NoC due to this minimum buffer.

Packet latency is also as important as throughput in networks. The longest waiting time of packet delivery is unacceptable in CHIPPER based techniques. In the worst case, the unlucky packet has to wait for its golden epoch for a complete cycle in CHIPPER based techniques. We propose a bus based technique to minimize the worst-case waiting time. We have analyzed both mesh and torus topologies with the proposed technique. When concepts are similar to both mesh and torus only mesh is considered in this paper to avoid repetition. When differences arise the two are dealt separately.

This paper is organized as follows. Section 2 analyzes the worst-case waiting time of a packet to get its golden epoch in CHIPPER. In Section 3 the proposed bus based method is given to reduce the worst-case waiting time. Experimental results are presented in Section 4 for both mesh and torus topologies. Section 5 has the two limitations of proposed method. Finally the conclusion is given in Section 6.

2. Analysis of Clock Based Synchronization

It is assumed that the NoC is clock accurate, $N \times N$ mesh/torus topology and a packet has “ k ” flits. In the worst case, at the beginning of the golden epoch the golden packet is in top left (or bottom left) router and the destination is the bottom right (or top right) of the mesh network. To reach the destination the packet has to cross $(N - 1)$ horizontal links and $(N - 1)$ vertical links. Let us have the worst-case scenario. Exactly at the beginning of golden epoch the processor connected to the corner source router starts to inject flits to the destination processor which is connected to the diagonally opposite corner. In this scenario, first flit needs $(2N - 2)$ clock cycles. The remaining $(k - 1)$ flits need $(k - 1)$ clock cycles to reach the destination. Hence the number of clock cycles needed for the golden epoch of a mesh is given by the following equation:

$$(2N + k - 3). \quad (1a)$$

In the case of torus, the maximum distance between two routers is “ N ” for even values of N and “ $N - 1$ ” for odd values of N . Hence the golden epoch of torus can be given by the following equations:

When N is even, one has

$$(N + k - 1). \quad (1b)$$

When N is odd, one has

$$(N + k - 2). \quad (1c)$$

The additional end to end router links of torus topology reduce the golden epoch significantly.

All routers have clock as one of the inputs. They have to load this count in the clock timer at the beginning of current golden epoch. It is decremented with clock. Once the count is zero, again the same count is autoreloaded and the next golden epoch begins.

This mechanism ensures live lock freeness. But we need to consider three points.

2.1. Golden Packet Is Consumed before the End of Golden Epoch. Here we assume the golden packet is present in the network during the golden epoch. The golden packet should be in one end and the destination should be in the other end to completely utilize the golden epoch. This does not happen always. Almost in all golden epochs, some cycles are run without golden packet. Let us analyze the probability of complete utilization of golden epoch.

To utilize all clock cycle of a golden epoch by golden packet, the following conditions should be satisfied:

(A-1) The processor which is connected to the sender has to inject first flit of the golden packet exactly at the beginning of the golden epoch.

(A-2) The sender should be a corner router and the destination should be the diagonally opposite corner router in a mesh topology. In torus topology, it has to be a corner router when the sender is center router or center router when the sender is a corner router.

First let us analyze the probability for condition (A-1). Here we have three cases:

(A-1.1) The golden packet is injected exactly at the first clock cycle of the golden epoch.

(A-1.2) The golden packet has been injected before the corresponding golden epoch. In other words, the golden packet is in the network when the golden epoch begins.

(A-1.3) The golden packet is yet to be injected during the golden epoch. In other words, the golden packet is not in the network. This is considered in case of Section 2.2.

Let us calculate the injection probability of (A-1.1) by considering equal probabilities for all clock periods of golden epoch. Suppose the golden epoch has “ m ” clock periods starting from 1 to m . The golden packet should be injected in the network exactly at the first clock cycle.

The probability for case (A-1.1) is $1/m$. (If we consider that equal probabilities for a packet are in the network and not in the network, the probability is reduced to $1/(2 \times m)$.)

Since the probability for not being in (A-1.1) is $(m - 1)/m$, without loss of generality let us assume that the probabilities for (A-1.2) and (A-1.3) are 50 percent of it (i.e., $(m - 1)/(2 \times m)$). In case (A-1.2), part or all or no flits can be consumed at the commencement of golden epoch. To utilize the complete golden epoch clock cycles, no flits have been consumed before the beginning of golden epoch. When this assumption is satisfied, the probability for case (A-1.2) is $(m - 1)/(2 \times m)$. (If the assumption is not satisfied the probability is reduced.)

Complete utilization of golden epoch is not possible in the case of (A-1.3). Therefore it is not analyzed now. This is considered in case of Section 2.2.

Now let us calculate the probability for condition (A-2). First consider an $N \times N$ mesh topology, with N^2 routers and $2 \times N \times (N - 1)$ edges. Here we have two cases:

(A-2.1) It follows case (A-1.1). The assumption is that golden packet is injected during the golden epoch.

(A-2.2) It follows case (A-1.2). The assumption is golden packet is available in the network and no flit has been consumed yet.

To utilize all $(2N + k - 3)$ clock cycles of golden epoch the distance between source router and destination router has to be network diameter (i.e., $(2N - 2)$). To satisfy this, the source router should be in one corner and the destination router should be in the diagonally opposite corner of the mesh topology. Since there are N^2 routers and 4 possible routers

for source and one possible router for destination after fixing source router, the probability for case (A-2.1) is $4/N^4$.

The total number of edges in mesh topology is $2 \times (N^2 - N)$. If the packet has single flit, to satisfy (A-2.2), the flit has to be any one of the chosen eight links and the destination has to be the diagonally opposite router. The probability for this is $(8/(2N^2 - 2N)) \times (1/N^2) = 4/(N^4 - N^3)$. If the packet has two flits, they have to be in both input links of a corner router. There are only 8 chances among $2 \times (N^2 - N)P_2$. Since it is very low, it is considered as the probability for case (A-2.2) which is $4/(N^4 - N^3)$. This is applicable only when the flit size is either one or two. For flits greater than this size case (A-1.2) does not support complete golden epoch utilization. For example, when flit size is 3 and network diameter is 8, the golden epoch is 10. In the worst case two flits are in the input links of a corner router. They will be consumed in eighth and ninth clock cycles. Tenth clock cycle is free of golden packet.

Therefore in an $N \times N$ mesh network, the probability for a k -flit ($k \neq 1, 2$) golden packet to use the entire golden epoch is

$$\frac{4}{m \times N^4}. \quad (2a)$$

When the flit size is either 1 or two, then the probability is

$$\frac{4}{m \times N^4} + \frac{2 \times (m - 1)}{m \times (N^4 - N^3)}, \quad (2b)$$

where $m = (2N + k - 3)$.

Now let us analyze torus topology. $N \times N$ torus topology has N^2 routers and $2 \times N^2$ edges. Hence the probability for case (A-2.1) is $8/N^4$, the probability for case (A-2.2) is $((8 \times {}_4P_k) / (2 \times N^2 P_k)) \times (1/N^2)$ and it is valid for flit sizes less than or equal to four. (In the above equation, ${}_nP_k = n! / (n - k)!$)

Therefore in an $N \times N$ torus network, the probability for a k -flit ($k \geq 5$) golden packet to use the entire golden epoch is

$$\frac{8}{m \times N^4}. \quad (2c)$$

When the flit size k is less than five, the probability is

$$\frac{8}{m \times N^4} + \frac{4 \times (m - 1)}{N^2 \times m} \times \frac{{}_4P_k}{2 \times N^2 P_k}, \quad (2d)$$

where “ m ” is given in (1b) and (1c).

Figures 2(a) and 2(b) show the probability of entire golden epoch utilization for mesh and torus for various N values. Though the utilization percentage of torus is more than mesh, the probability of complete utilization of entire golden epoch in both mesh and torus is close to zero and is ignorable. The packet has been consumed before the end of the golden epoch (provided the golden packet is available in the network when golden epoch begins) with 0.99 probability.

Figure 3 shows the probability of various percentage of the golden epoch utilization. From the figure it is shown that the probability of golden epoch clock cycles without golden packet is significant. The oscillation in the usage of torus is because of the variation of golden epoch clock cycles in even and odd number values of N .

2.2. The Golden Epoch without Golden Packet. There are chances that the current golden packet might be consumed already. Similarly there are chances that the current golden packet is yet to be injected in the network. In these two cases, a complete golden epoch is run without golden packet. This probability depends on factors such as injection rate of packets and network traffic. Without loss of generality one can assume that the golden packet is not in the network during its golden epoch with 50 percent of probability. This percentage will be reduced rapidly when injection rate is high and the network is in saturated condition. In this scenario the packet encounters many deflections. Conversely, this percentage is increased steeply when injection rate is low or the traffic is light since in this case the number of deflections is minimal. On the average the probability of golden epoch without golden packet can be assumed as 0.5 without loss of generality.

2.3. A Fraction of the Golden Packet Is Not Delivered. The golden packet is yet to be injected at the beginning of golden epoch. Some cycles of golden epoch were over. Now the golden packet is injected in the network. Remaining cycles of golden epoch are insufficient to completely deliver this packet. The probability for this scenario is very less.

The first two cases show that an unnecessary delay is incurred to choose the next golden packet. Cumulatively this increases the time to delivery of other live locked packets. The third point shows that, in the worst case, the remaining flits are live locked for a complete cycle. Similarly the packets which have missed their golden epoch also wait for a complete cycle. The worst-case time of complete cycle is equal to the product of number of routers, maximum number of packets injected by a router, and the number of clock periods in the golden epoch. The following equation gives the worst-case total number of clock cycles present in the complete cycle:

$$N^2 \times 2^m \times (D + k - 1), \quad (3)$$

where N is the number of routers present in row/column, m is the number of bits allocated for packet identification, D is the diameter of the network, and k is the number of flits in a packet.

For an 8×8 mesh network with 8 bits allocated to the packet id field (256 packets by a source) the worst-case complete cycle time has $(64 \times 256 \times (14 + 3)) = 278528$ clock cycles whereas an 8×8 torus network with the same specifications has 180224 clock cycles.

The destination processor has to wait almost 3 million clock cycles in the case of mesh and almost 2 million clock cycles in the case of torus to obtain the message after the injection into the network. Though the packet is finally considered as a golden packet and is delivered to destination, it is not an acceptable latency for a packet. We propose to reduce this latency by eliminating the golden epoch clock cycles which have no golden packets.

FIGURE 2: (a) Probability chart of full utilization of golden epoch for $N \times N$ mesh. (b) Probability chart of full utilization of golden epoch for $N \times N$ torus.

FIGURE 3: (a) Probability chart of more than 60 percent utilization of golden epoch for $N \times N$ mesh. (b) Probability chart of more than 80 percent utilization of golden epoch for $N \times N$ torus.

3. Proposed Technique: Broadcast Bus (BBUS) Based Synchronization

The basic requirement is all routers should know about golden epoch cycles without golden packets. When they all have this information, they all can terminate the current golden epoch and begin the next golden epoch simultaneously. Since bus is the best broadcast medium, we propose to use bus for broadcasting the termination of current golden epoch. We divide the analysis into asynchronous bus and synchronous bus. The algorithms are slightly modified according to the nature of buses. The concepts are the same for mesh and torus methodology. To avoid repetition only mesh is considered in this section.

3.1. Usage of Asynchronous Broadcast Bus. The suggestion is inclusion of a single broadcast bus with the available architecture as shown in Figure 4. This bus has been pulled up to logic “1” by a pull-up resistor in normal conditions. Any router can place logic “0” on this bus. Since it is a strong “0” and “1” is weak, the bus status is altered to “0.” Once the placement of logic “0” is stopped by the router, the

bus goes back to its original logic “1” condition. With this understanding we present the algorithms for the following two cases:

- (A) Golden packet is not in the network.
- (B) Golden packet is consumed before the completion of the gold period.

(A) Analysis of Network without Golden Packet. All routers check their input links for the golden packet. Those routers which have it in their input link modify the status of bus by placing logic “0” on the bus and continue with that golden epoch. Those routers which do not have the golden packet in their input link observe the broadcast bus input for a stipulated time. If the status of the bus is going down within the stipulated time then they continue with the current golden epoch. Otherwise they terminated the golden epoch after the stipulated time and proceed with the next golden epoch in a synchronized manner.

The stipulated time is decided by the information travelling time on bus from one end of network to another end, for example, the travelling time from top left to bottom right in

FIGURE 4: 4×4 mesh NoC with BBus.

a mesh topology. This might be less than, equal to, or greater than one or more clock periods of network. If this time is less than or equal to “ n ” clock periods of network, then “ n ” clock period is the stipulated time. If this time is greater than “ n ” clock periods, then “ $n + 1$ ” is the stipulated time. A golden epoch begins in a rising or falling edge of clock period in a cycle accurate network.

(B) *Golden Packet Is Consumed before the Completion of the Gold Period.* The destination router has to inform this to the remaining routers and requests for the termination of current golden epoch. In one gold period, all routers have the information of source id and packet id of the golden packet. But they do not know the destination id of the packet. Depending on their location in the network and the destination router which broadcasts the message, different routers receive it at different times. If the receiving router has the spatial information of destination router in the network then it knows when it will receive the message and when the router which is placed far away from the transmitting router will receive the message. If all routers have this knowledge then they take the action with synchronization. Only when the end to end broadcast time is within a clock period can synchronization be achieved with the clock period. Otherwise synchronization cannot be achieved.

This can be solved in two ways. One way is the addition of buses to code the destination id. The usual broadcast message is sent along with the id of the broadcasting router. Since the routers know the space of destination router in the network they know how much more time the message will take to reach the farthest router. The routers in different place in the network have to wait for different time before they synchronously decide the next golden epoch.

The second way is the inclusion of a hardware module and only one more bus. We name the hardware component as golden period terminator (GPT). All routers have one in-bus and one out-bus. Out-bus from routers is connected to the in-bus of GPT and the in-bus to routers is coming from out-bus

FIGURE 5: 4×4 mesh NoC with BBus and GPT.

of GPT. The broadcasting router has to send the message to GPT which will relay the information to all routers in the network. In this way the router which is sending the message also gets the message from its in-bus. Only the destination router of golden packet is altering the status of out-bus. So, there is no collision. Only GPT is changing the status of in-bus of routers. There is no collision. Since all routers know the position of GPT in the network, the next golden epoch is decided in synchronized manner.

We prefer the second method since the area complexity is reduced compared to the first method by the reduction of bus width. The power consumption also reduced since the data on the bus has to be changed almost in all golden epochs. The position of GPT is very importantly related to performance. If it is placed in one end, then the round about time is twice of the end to end broadcasting time. We prefer to place it in center as shown in Figure 5 in both mesh and torus topology. In torus the round about time is twice of the end to end broadcasting time.

```

(1) if ( $R_p \neq G_p$ )
(2) {
(3)   while ( $!t_s$ ) loop
(4)   {
(5)     Observe  $b_{ri}$ ;
(6)   }
(7)   if ( $b_{ri} = "0"$ )
(8)   {
(9)     Continue with  $C_{ge}$ ;
(10)  }
(11)  else
(12)  {
(13)    Terminate  $C_{ge}$ ;
(14)  }
(15) }
(16) else
(17) {
(18)    $b_{ro} = "0"$ ;
(19) }

```

ALGORITHM 1: Golden epoch without golden packet: function of all routers in asynchronous BBUS.

```

(1) While (1) loop /* endless loop */
(2) {
(3)   While ( $!t_s$ ) loop
(4)   {
(5)     observe  $b_{gi}$ ;
(6)     if ( $b_{gi} = "0"$ )
(7)     {
(8)        $b_{go} = "0"$ ;
(9)       Exit from loop;
(10)    }
(11)  }
(12)  if ( $b_{go} = "0"$ )
(13)  {
(14)    while ( $!t_s$ ) loop
(15)    {do nothing
(16)    }
(17)     $b_{go} = "1"$ ;
(18)  }
(19) }

```

ALGORITHM 2: Golden epoch without golden packet: function of GPT in asynchronous BBUS.

Algorithms 1, 2, 3, and 4 explain the above concepts. We include GPT for the first case also. The following symbols are used in algorithms:

R_p : packet that is available in the in-link of router.

C_{ge} : current golden epoch.

G_p : golden packet.

t_s : predetermined stipulated time.

b_{ro} : out-bus of router.

b_{go} : out-bus of GPT.

```

(1) if ( $R_p \neq G_p$ )
(2) {
(3)   if ( $!(\text{contention for } P_p)$ )
(4)   {
(5)      $P_p = F_{re}$ ;
(6)   }
(7)   else
(8)   {
(9)      $P_p = F_{ra}$ ;
(10)  }
(11) }
(12) else
(13) {
(14)   if ( $!(\text{all flits ejected})$ )
(15)   {
(16)     Eject flits;
(17)   }
(18)   else
(19)   {
(20)     if (the remaining period > broadcasting time)
(21)     {
(22)        $b_{ro} = "0"$ ;
(23)     }
(24)   }
(25) }

```

ALGORITHM 3: The golden packet is consumed well before the end of golden epoch: function of receiver router in asynchronous BBUS.

```

(1) if ( $R_p = G_p$ )
(2) {
(3)    $P_p = F_g$ ;
(4) }
(5) else
(6) {
(7)   if ( $b_{ri} = "1"$ )
(8)   {
(9)     if ( $!(\text{contention for } P_p)$ )
(10)    {
(11)       $P_p = F_{re}$ ;
(12)    }
(13)    else
(14)    {
(15)       $P_p = F_{ra}$ ;
(16)    }
(17)  }
(18)  elseif ( $b_{ri} = "0"$ )
(19)  {
(20)    while ( $!t_s$ ) loop
(21)    {do nothing;
(22)    }
(23)    Terminate  $C_{ge}$ ;
(24)  }
(25) }

```

ALGORITHM 4: The golden packet is consumed well before the end of golden epoch: function of intermediate routers in asynchronous BBUS.

b_{ri} : in-bus of router.

b_{gi} : in-bus of GPT.

P_p : productive port.

F_g : golden flit.

F_{re} : flit which request P_p .

F_{ra} : random flit among the competing flits.

b_{ro} is connected to b_{gi} and b_{ri} connected to b_{go} . t_s is varying with respect to the router positions. The router which is closely placed to GPT has highest t_s and the one which is farthest away has lowest t_s .

Algorithm 1 gives the functionality of all routers. All routers check their input links at the beginning of golden period. If golden packet is available in input link, this is conveyed to GPT by placing logic "0" on out-bus of router. Else they observe their in-bus for stipulated time. If the in-bus status is altered, the current golden epoch is continued. Else the current golden epoch terminated.

GPT function is given in Algorithm 2. GPT watches its in-bus for the stipulated time. If the bus is zero within the time, it will broadcast this information through out-bus of GPT. After a stipulated time out-bus of GPT comes back to its original logic "1" status.

3.2. Usage of Synchronous Broadcast Bus. Synchronous broadcast bus differs from the previous technique by the fact that the bus is synchronized with clock. The changes in bus may be made between clock ticks. But the actions based on the changes are taken only in the beginning of the next clock. Let " t_1 " be the time of modification of bus status by one or more routers and let " t_2 " be the time that the modification reached all other routers in the network. The basic assumption in this scheme is that the time difference between " t_2 " and " t_1 " is always less than one clock period of the network. If it is greater than that then $(t_2 - t_1)$ is the critical time and clock period should be at least equal to this time. No GPT is needed in this technique since wherever the broadcasting router is placed all routers can get the message before the start of next clock. Hence the action can be taken in the next clock edge. We present the modified Algorithms 5, 6, and 7. The modification is only in t_s . Routers wait for next clock edge.

Only destination router alters the status and intermediate routers observe the status in each and every clock. Hence there is no collision in the bus. Let us assume the golden epoch has " p " clock periods. Destination router receives all flits at " p_1 " clock cycle. Approximately $(p - p_1)$ clock cycles are idle if the synchronization is provided by clock counter. In this mechanism those clock cycles are used for next golden epoch. Hence waiting time of other live locked packets is decreased.

4. Experimental Results and Analysis

The network parameters to evaluate the performance enhancement are given in Table 1. VHDL is used to

```

(1) if ( $R_p = G_p$ )
(2) {
(3)    $b_{ro} = "0"$ ;
(4)   Continue  $C_{ge}$ ;
(5) }
(6) else
(7) {
(8)   Listen  $b_{ri}$  for a clock period
(9)   if ( $b_{ri} = "0"$ )
(10)  {
(11)    Continue  $C_{ge}$ ;
(12)  }
(13)  else
(14)  {
(15)    Terminate  $C_{ge}$ ;
(16)  }
(17) }
```

ALGORITHM 5: Golden epoch without golden packet: function of all routers in synchronous BBUS.

```

(1) if ( $R_p \neq G_p$ )
(2) {
(3)   if (!contention for  $P_p$ )
(4)   {
(5)      $P_p = F_{re}$ ;
(6)   }
(7)   else
(8)   {
(9)      $P_p = F_{ra}$ ;
(10)  }
(11) }
(12) else
(13) {
(14)  if (!(all flits ejected))
(15)  {
(16)    Eject flits;
(17)  }
(18)  else
(19)  {
(20)     $b_{ro} = "0"$ ;
(21)    Wait for next clock;
(22)    Terminate  $C_{ge}$ ;
(23)  }
(24) }
```

ALGORITHM 6: The golden packet is consumed well before the end of golden epoch: function of receiver router in synchronous BBUS.

implement the network and Xilinx Virtex-5 XC5VLX50T FPGA is used to implement the design.

The packet number is represented by 4 bits. After all "1" combination, the next packet will be numbered as "0" again. Source and destination addresses have 6 bits each; 3 bits is representing row number and the remaining 3 bits is column


```

(1) if ( $R_p = G_p$ )
(2) {
(3)    $P_p = F_g$ ;
(4) }
(5) else
(6) {
(7)   if ( $b_{ri} = "1"$ )
(8)   {
(9)     if (! (contention for  $P_p$ ))
(10)    {
(11)       $P_p = F_{re}$ ;
(12)    }
(13)   else
(14)   {
(15)      $P_p = F_{ra}$ ;
(16)   }
(17) }
(18) elsif ( $b_{ri} = "0"$ )
(19) {
(20)   Wait for next clock;
(21)   Terminate  $C_{ge}$ ;
(22) }
(23) }

```

ALGORITHM 7: The golden packet is consumed well before the end of golden epoch: function of intermediate routers in synchronous BBUS.

TABLE 1: Network parameters for experimental setup.

Network parameters	Values
Topology	8×8 mesh, 8×8 torus
Flits/packet	4
Packet/source	16
Source and destination selection	Pseudorandom numbers based on uniform distribution
Router buffer size	Zero
Deflection count bits	8 bits
Broadcast bus size	2 bits
Link size	32 bits

numbers in the 8×8 matrix. Link size is the bus size between adjacent routers.

The simulation results are shown in Figure 6. Existing CHIPPER and the proposed synchronous BBUS methods are compared for the golden epoch rate, golden flit ejection rate, latency, and throughput.

As shown in the result, the golden epoch rate is independent of injection rate. In mesh, CHIPPER produces only one golden epoch in 18 clock cycles and in torus it produces one golden epoch in 12 clock cycles. But in BBUS the golden epoch rate is almost one per clock cycle when injection rate is low. This increases the possibility of identifying more golden flits than the conventional CHIPPER as shown in Figure 7.

Figure 7 shows the evaluation results of golden flit rate of existing and proposed methods. As expected the proposed

FIGURE 6: Estimation of golden epoch rate for mesh/torus.

mechanism delivers more golden flits. After applying the proposed BBUS technique the golden flit rate is almost twice of CHIPPER in mesh topology. Torus delivers more golden flit in CHIPPER compared to mesh. From the experimental results it is shown that BBUS technique further enhances the performance of torus. Note that the difference is reduced significantly if injection rate increases. During high injection rate traffic is increased and it significantly increases the probability of deflection. Due to this the throughput is decreased. As more packets are delivered only during their golden epoch, the golden flit delivery rate is increased when injection rate increases. Since more packets are turned as golden packets, the golden epoch cannot be terminated until the packet is delivered. So, the golden epoch rate is decreased in BBUS and the difference between the two techniques are reduced as shown in Figures 6 and 7.

Figure 8 analyzes the flit rate with more than half diameter of network. When injection rate is low, almost all flits are consumed before their golden epoch comes. Due to less traffic almost no packet is deflected or the deflection number is very less. The deflection count is greater than half diameter of the network only on rare occasion. These unlucky packets usually wait for their golden epoch. Since BBUS quickly announces these packets as golden packet due to very high golden epoch rate compared to CHIPPER, the rate of such packets is almost zero in BBUS. When injection rate increases the rate of such packets is increasing exponentially in both techniques. But as shown in the results, BBUS maintains its superior performance.

5. Limitations of BBUS

Throughput is the number of packets delivered per unit time. In BBUS it is slightly less than CHIPPER when the injection rate is less than 0.5 flit/cycle/node as shown in Figure 9. It is due to the fact that golden packets make the other packets deflect. Since BBUS delivers more golden flits, the competition to other packets is also increased. When the injection rate is greater than 0.5 flit/cycle/node, then the throughput is the same with CHIPPER with the advantage of maintaining the low average latency of packets.

The second limitation is area consumption. In asynchronous BBUS, due to the addition of two bus lines and GPT module the area is increased around 6 percent for a 32-bit bus between routers. It is decreased when the bus

FIGURE 7: (a) Ejection rate of golden flit in mesh. (b) Ejection rate of golden flit in torus.

FIGURE 8: (a) Latency analysis for mesh. (b) Latency analysis for torus.

FIGURE 9: (a) Throughput analysis for mesh. (b) Throughput analysis for torus.

size is increased. The overhead is decreased to 1.8 percent for 128-bit bus between routers. Hence this limitation is not predominant when the link size is at least 128 bits. Most of the NoC circuits use wider link size.

Since the throughput difference is not severe and the average latency difference is increased tremendously it is always better to use the proposed method when the nodes

need multiple packets before the commencement of data processing.

6. Conclusion

In this paper a bus broadcasting approach is used in CHIPPER base NoC. It is shown that the area and throughput

ratings are almost similar to CHIPPER. But the longest waiting time of packets is tremendously minimized in the proposed method compared to CHIPPER. The worst-case comparison is done intuitively in the following way. If a packet misses its golden epoch in the proposed method, it has to wait for a complete cycle time. For example, consider a generic $N \times N$ mesh network. The nodes are injecting packets with packet_id “0” to “ $p-1$.” A packet is split into “ F ” flits. In both methods the worst-case time for a flit to get its golden epoch is calculated as clock cycles as given below:

$$N^2 \times p \times \text{Golden epoch period.} \quad (4)$$

The golden epoch period is $2 \times N + k - 3$ for mesh and $N + k - 1$ (or) $N + k - 2$ times for torus. In this waiting period, CHIPPER may deliver zero to N^2 golden packets. If this is the waiting period, BBUS always delivers N^2 golden flits. If the rate of golden flit delivery is decreased, then the waiting time is also decreased in BBUS. If no golden flit is delivered, then the worst-case waiting time is decreased to $N^2 \times p$ clock cycles in BBUS. It is golden epoch period times more superior than CHIPPER.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

References

- [1] W. J. Dally and B. Towels, “Route packets, not wires: on-chip interconnection networks,” in *Proceedings of the Design Automation Conference (DAC '01)*, pp. 684–689, Las Vegas, Nev, USA, June 2001.
- [2] L. Benini and G. De Micheli, “Networks on chips: a new SoC paradigm,” *Computer*, vol. 35, no. 1, pp. 70–78, 2002.
- [3] S. Kumar, A. Jantsch, J.-P. Soininen et al., “A network on chip architecture and design methodology,” in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, pp. 105–112, Pittsburgh, Pa, USA, April 2002.
- [4] L. Benini and D. Bertozzi, “Network on chip architectures and design methods,” *IEEE Proceedings—Computers and Digital Techniques*, vol. 152, no. 2, pp. 261–272, 2005.
- [5] W.-C. Tsai, Y.-C. Lan, Y.-H. Hu, and S.-J. Chen, “Networks on chips: structure and design methodologies,” *Journal of Electrical and Computer Engineering*, vol. 2012, Article ID 509465, 15 pages, 2012.
- [6] W. J. Dally and B. Towels, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann Publishers, 2003.
- [7] P. Ezhumalai, A. Chilambuchelvan, and C. Arun, “Novel NoC topology construction for high-performance communications,” *Journal of Computer Networks and Communications*, Article ID 405697, 6 pages, 2011.
- [8] Z. Lu and A. Jantsch, “TDM virtual-circuit configuration for network-on-chip,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 8, pp. 1021–1034, 2008.
- [9] T. T. Ye, L. Benini, and G. De Micheli, “Analysis of power consumption on switch fabrics in network routers,” in *Proceedings of the 39th Annual Design Automation Conference (DAC '02)*, pp. 524–529, IEEE, New Orleans, La, USA, June 2002.
- [10] C. A. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, “ViChaR: a dynamic virtual channel regulator for network-on-chip routers,” in *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '06)*, pp. 333–346, Orlando, Fla, USA, December 2006.
- [11] Y. Zhang, R. Morris Jr., and A. K. Kodi, “Design of a performance enhanced and power reduced dual-crossbar Network-on-Chip (NoC) architecture,” *Microprocessors and Microsystems*, vol. 35, no. 2, pp. 110–118, 2011.
- [12] T.-Y. Lee and C.-H. Huang, “Design of smart power-saving architecture for network on chip,” *VLSI Design*, vol. 2014, Article ID 531653, 10 pages, 2014.
- [13] T. Moscibroda and O. Mutlu, “A case for bufferless routing in on-chip networks,” in *Proceedings of the 36th International Symposium on Computer Architecture (ISCA '09)*, pp. 1–12, Austin, Tex, USA, June 2009.
- [14] P. Baran, “On distributed communications networks,” *IEEE Transactions on Communications Systems*, vol. 12, no. 1, pp. 1–9, 1964.
- [15] A. Monemi, C. Y. Ooi, and M. N. Marsono, “Low latency Network-on-Chip router microarchitecture using request masking technique,” *International Journal of Reconfigurable Computing*, vol. 2015, Article ID 570836, 13 pages, 2015.
- [16] J. Lin, X. Lin, and L. Tang, “Making-a-stop: a new bufferless routing algorithm for on-chip network,” *Journal of Parallel and Distributed Computing*, vol. 72, no. 4, pp. 515–524, 2012.
- [17] C. Fallin, G. Nazario, X. Yu, K. Chang, R. Ausavarungnirun, and O. Mutlu, “MinBD: minimally-buffered deflection routing for energy-efficient interconnect,” in *Proceedings of the 6th IEEE/ACM International Symposium on Networks-on-Chip (NoCS '12)*, pp. 1–10, Lyngby, Denmark, May 2012.
- [18] C. Fallin, C. Craik, and O. Mutlu, “CHIPPER: a low-complexity bufferless deflection router,” in *Proceedings of the 17th International Symposium on High-Performance Computer Architecture (HPCA '11)*, pp. 144–155, San Antonio, Tex, USA, February 2011.
- [19] I. Z. Stojanovic, M. D. Jovanovic, and G. L. Djordjevic, “Low-cost port allocation scheme for minimizing deflections in bufferless on-chip networks,” in *Proceedings of the 21st Telecommunications Forum (TELFOR '13)*, pp. 357–360, IEEE, Belgrade, Serbia, November 2013.

Hindawi

Submit your manuscripts at
<http://www.hindawi.com>