© 2017 IJSRSET | Volume 3 | Issue 3 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Low-Latency Communication Architecture For MACS

Bhagyamma S*[1], Neelappa[2], Rajesha K S[3], Shashikanth N S [4], Ranjitha B C[5], Anuradha K S[6]
[*1,2,3,4,5,6]Department of Electronics & Communication Engineering, GEC, Kushalnagar, Kodagu, Karnataka, India

ABSTRACT

To enhance the parallelism and system scalability for single chip VLSI design, NoC's are used and they are increasing day by day. In this paper we proposed MACS architecture which provides low latency, flexibility and scalability using a circuit switching technique. MACS enhances their inter processing element communication to maximize bandwidth utilization. The proposed architecture has been verified and tested on FPGA board and the results revels that the proposed architecture has better performance in terms of speed and power with compromising the area. The FPGA implementation of MACS requires 440 slices with frequency of 152.09MHz and power consumption of 0.457W.

Keywords: NoC, minimal adaptive, distributed round-robin, FPGA, HDL

I. INTRODUCTION

In past decade, most of computer components integrated in single silicon chip i.e. system on chip (SoC). It has contained powerful processors, analog, digital and mixed signal components on a silicon substrate. SoC mainly used in embedded applications. The foremost problem of these components is communication between them [1]. If the number of components has increased, communication between components has considerably complex. Communication networks are connecting different geographically distributed points. In point to point communication, the connection of the any two resources is fixed. It provides flexibility with avoiding arbitration but some resources have not involved for processing data. It had been leading a problem of less utilized resources while to increase resource utilization, bus based on chip communication has introduced. In bus technology, components are connected with a single bus. It is simple and widely used as components are connected through bus. Due to a single bus, servicing of components has been delayed and consumed much power. It is not scalable as the bandwidth has shared to entire system resources. To overcome the problem, hierarchical bus has used. Bridge has attached between the multiple buses and saved the power consumption due to the not using long wires. This technique having multiple buses which leads arbitration between buses and numerous wires connected to the bus. To solve arbitration overhead, bus matrix method has introduced. The resources are connected in a matrix manner. By using matrix bus, the arbitration has overcome but due to on chip interconnects electric noise, degrading performance, energy consumption and scalability increased. These problems have been given a new paradigm i.e. NoC. Dally [2] and Benini [3] have introduced this new on chip communication for SoC. NoC is the system which consists of a group of routers with processing elements (PEs) and formed a topology (ex. Mesh, torus, folded torus, tree/fat tree and ring). While transferring data, source router connects other router through PE until it reaches to its destination router. PEs has controlling the data transmission over the flits or dedicated channels. NoC has been giving notable improvements over the conventional bus and the sharing of NoC interconnects helped to reuse of the resources. NoC is presented in various journals, special conferences and NoC symposium [4].

The traditional NoC routers are composed of structural and transmission parameters. The structural parameters are included arbiters, buffers and routing algorithms while transmission parameters are channel allocation and switching mode. A typical router is having five ports to handle the information. The five ports of NoC are North port, East port, South port, West port and Local port. Each port are having bi-direction channel totransmit/receive the data from port to port. The router has interconnected its own PE through local port and network interface (NI).

II. RELATED WORK

Wiklund et. Al [1] proved that the mesh topology was the most appropriate for on-chip network. Wilkund and zhong both provided strong arguments for the advantages of circuit switched NoC's over packet switched NoC's. Wilkund [1] proposed a mesh topology NoC using distributed arbitrations a new switched technique as packet connected circuits. Zheng proposed a TDM. RMBOC was implemented a 2D mesh topology HiHon presented the programmable NoC. Carara et. Al[7] proposed a 4x4 NoC mixed switching techniques with fixed sized packets and two parallel interface.

III. NoC Architecture

3.1 NOC ROUTER ARCHITECTURE

The NoC has made up of following building blocks. Those are topology, switching, routing and crossbar. A typical NoC has bidirectional inputs and outputs (N, S, W, E and L). The data is transferred using bi-directional channels either the local port to router or router to router. The bidirectional local port is connected to its PE through NI. The PE has process the data information and transferred through the fixed channel. Each input port having a buffer to store the input data which come from the output port. The buffers have worked as First In First Out (FIFO) and shown in Figure 1. These buffers are having association with input and output ports to store the data. The buffer size of input port has depended on the packet size of the data. The arbitration unit controls the input and output ports among number of available ports. The routing unit is responsible for the how packets are routed to the source to destination router. By using of routing and arbitration units the crossbar switch transfer the data from input port to the output port. If the requested is buffer full in destination port then the transferring packet has to wait until the buffer has become free.

To avoid the errors of NoC, use the virtual channel in association with physical channel. These virtual channels provide the extra channels to moving the packets from one port to another and deadlock can be avoided. Virtual channel controller is needed to organize the virtual channel with physical channel.

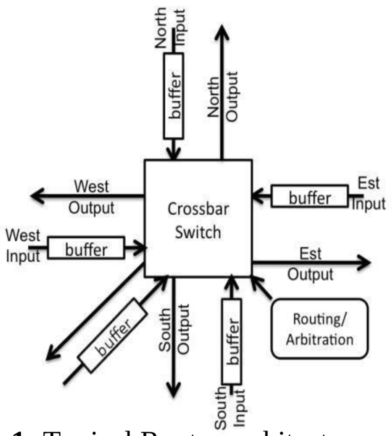

Figure 1: Typical Router architecture with five input and output ports (north, east, south, west and local).

3.2 MACS ARCHITECTURE

Figure 2. (left) depicts a 3x3 MACS-NoC's high-level architectural layout. X and Y coordinates identify individual switch addresses based on horizontal and vertical positions, respectively. Each switch's two PEs are addressed relative to the PE's connected switch. MACS has four total switch ports with one port connected to each neighbouring switch (left, right, up, and down) and two local ports connected to the two PEs (Figure 2. (right)). A port identification number (PID) uniquely identifies each port. Each port contains multiple input and output communication lanes to support multiple simultaneous communication channels between different PE pairs (denoted by the K tunable architectural parameter in Figure 2.). Furthermore, each lane contains input and output data signals (denoted by the W tunable architectural parameter in Figure 2.) to provide data transfer bandwidth on a communication channel. MACS provides guaranteed throughput using a circuit-switched communication methodology with distributed arbitration. In this section, we provide an overview of switch operations followed by switch architectural details.

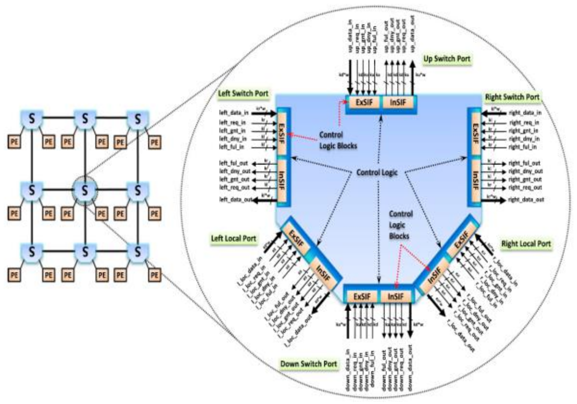

Figure 2: 3 x 3- MACS's architectural layout (left) and detailed MACS switches architecture (right).

3.2.1 SWITCH OPERATION

Switch operations include communication channel establishment for inter-PE data transfers (transactions), waiting for transaction completion, and releasing communication channel resources (e.g., logic elements, registers, etc.). Channel establishment connects an input lane to an output lane and is the process of allocating channel resources for routing incoming channel establishment requests and data on this input-output lane connection, thus establishing a dedicated communication channel. After channel resource allocation, the switch waits until transaction completion before releasing these resources. Each port contains control logic for channel establishment. The control logic consists of two types of control logic blocks: an External Signal Forwarder (ExSIF) and an Internal Signal Forwarder (InSIF) (collectively referred to as signal forwarders). Signal forwarders are responsible for controlling all communication operations such as request servicing, channel establishment negotiations, and channel release.

3.2.2 SWITCH ARCHITECTURE

Figure 3. (right) depicts MACS's detailed switch architecture including all switch/local ports and signals associated with each port's input and output lanes. Switch ports and local ports are similar in that the ports consist of the same set of lanes. A lane identification number (LID) uniquely identifies each port's lane. Lanes can be further classified as input and output lanes that carry request/data signals into and out of a switch, respectively. Each input lane consists of several control signals (req_in, gnt_out, dny_out, and ful_out) and W input data signals (data_in). Similarly, each output lane consists of several control signals (req_out, gnt_in, dny_in, and ful_in) and W output data signals (data_out). Control signals negotiate channel establishment while data signals provide inter-PE communication bandwidth (increasing W increases MACS's communication bandwidth). MACS's ports are highly parametric, providing fine grained per-direction communication bandwidth specialization. Each switch/local port can be specialized with different numbers of input or output lanes. K denotes a port's lane count, and to distinguish between distinct switch/local ports, K is appended with the PID. Thus, a switch has six total K parameters for communication bandwidth specialization (the number of simultaneous different per-port inter- PE communication channels, and thus the bandwidth, monotonically increases with K).

In order to control communication channels and ensure correct communication, switches record channel routing information in status registers. Channel routing information specifies lane availability and input-output port lane connections.

3.3 SIGNAL FORWARDER AND STATUS REGISTERS

Figure 3. depicts the internal connections between two arbitrary switch port lanes (connections with the remaining switch port lanes are labelled but not shown) and the status register placement. The signal forwarders, ExSIF and InSIF, establish internal connections between input and output lanes, respectively. For readability, Figure 3. has been subsisted to show components and signals for a sample input-output lane connection. In the actual architecture, every lane contains all shaded components. ExSIF and InSIF use the request register to forward, maintain, and release incoming requests from the neighbouring switches and the request's grant/deny to neighbouring switches, respectively. InSIF polls internal requests forwarded by ExSIF in a distributed round-robin fashion and assigns an output to a neighbouring switch based on the communication channel routing algorithm and the availability register. The availability register contains information about the number of output lanes not serving an output to a neighbouring switch. InSIF and ExSIF use the connectivity register to forward, maintain, and release internal requests forwarded by ExSIF and the request's grant/deny coming from the neighbouring switch, respectively. MACS's routing algorithm evaluates multiple routing paths and allows each port to store the port's unavailable communication resources (i.e., number of the port's output lanes that are reserved for established channels) in the availability register. As the number of unavailable communication resources decreases, the likelihood of a new channel establishment assigned to this port increases and thereby maximize bandwidth utilization.

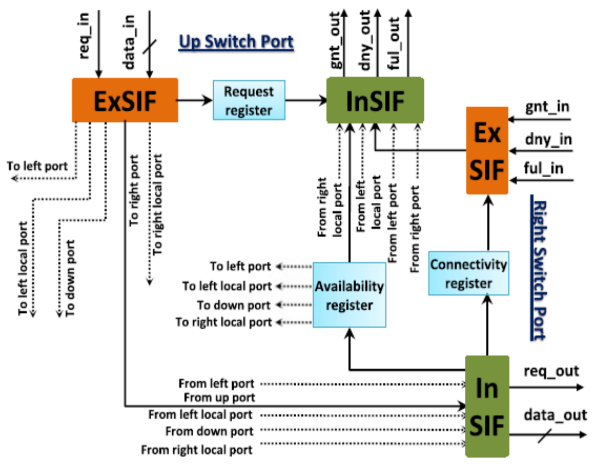

Figure 3: A subset of internal connection between the signal forwarders (ExSIF and InSIF) and status register placement for two arbitrary lanes (up and right).

As shown in Figure 3. an InSIF receives incoming requests from all ExSIFs, except InSIF's own port's ExSIF. Selecting and maintaining a set of incoming requests for a set of outputs requires a complex round-robin arbiter, which could increase the area and channel setup latency of the switch and reduce the switch's maximum attainable operating frequency. A typical round-robin arbiter (referred to as a typical arbiter henceforth) identifies an incoming request from a set of incoming requests in a round-robin fashion, identifies available output lane(s) in a round-robin fashion, and connects the incoming request to the output lane. Identification and connection of multiple incoming requests to multiple output requests simultaneously greatly increases the typical arbiter's complexity. In order to reduce the arbiter's complexity, the InSIF uses distributed round-robin arbitration with cyclic distribution of incoming requests over output lanes. This distribution causes disjoint sets of incoming requests to map to different output lanes. Additionally, if the K architectural parameter for all ports are equal, cyclic distribution ensures one-to-one correspondence between the LID of incoming requests' input lane and the LID of the output lane (e.g., all requests arriving on LID ¼ n are always mapped to the output lane with LID ¼ n). This one-to-one correspondence of LIDs provides improved controllability of the NoC due to constraints on routing traceability.

3.3 METHODOLOGY

A 3x3 Highly Customizable Low-Latency Communication Architecture for NoC's is proposed and designing. X and Y coordinates identify individual switch addresses based on horizontal and vertical positions, respectively. Each switch's two PEs are addressed relative to the PE's connected switch. MACS has four total switch ports with one port connected to each neighbouring switch (left, right, up, and down) and two local ports connected to the two PEs. A port identification number (PID) uniquely identifies each port. Each port contains multiple input and output communication lanes to support multiple simultaneous communication channels between different PE pairs. Furthermore, each lane contains input and output data signals to provide data transfer bandwidth on a communication channel. A Highly Customizable Low-Latency Communication Architecture provides guaranteed throughput using a circuit-switched communication methodology with distributed arbitration. In this section, we provide an overview of switch operations followed by switch architectural details.

The proposed BEDT is an advanced version of all existing NoC for high speed communications and it has both encoder and transmission modules at the transmitter and at receiver have decoder and its receiver modules and vice-versa. The BEDT has provided an efficient performance analysis like speed and power consumption for effective communications and consists of the following modules:

- ➢ Design of single switch NoC
- ➢ Novel Bit transition encoder
- ➢ Novel Bit transition decoder
- ➢ Hardware Implementation on FPGA
- ➢ Comparison of NoC BEDT with existing work

IV. DESIGN AND IMPLEMENTATION

4.1 DESIGN OF SINGLE SWITCH NoC

The NoC single switch is designed for the purpose of data transfer between source and destination at a transmission rate of 10GHz. The proposed switch consists of the two PEs, control circuit, First-In-First-Out (FIFO) and Finite State Machine (FSM). The two PEs have been performing the operation for two parallel data which are received from two different destinations.

Each PE has memory for storage and performing the operation at a rate of 10GHz i.e., each packet of data is transmitted at speed of $\frac{1}{10\text{GHz}}$ that means 0.1ns is the time required to transmit each packet. Therefore the proposed NoC switch has feasibility to access to different data from two different sources and both data are performed parallel.

The control unit having reset and selection control signals to control overall design, the reset is to reset all internal registers and selection is to select the direction of data transfer, there are four directions and one virtual channel, the direction are like north, south, west and east. Based on the selection signal the data will transfer from source to destination as shown in Figure 4.

When all direction lines are busy for data transmission, then the proposed switch automatically switches over to virtual channel for effective communication. This virtual channel concept is a costly and complex circuit, but it can be used for emergency data transmission.

Each switch has its one virtual channel and based on request command on selection line, the virtual channel will get enable and connect to destination.

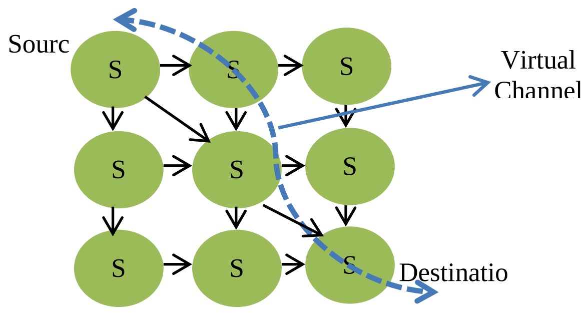

Figure 4, 3x3 NoC switch and their direction with virtual channel

4.2 TOPOLOGY

The topology is described as how ports/nodes are connected in a network. The topology has indicated the number of alternative routes between the ports and controlled the network contention along with different traffic patterns. The different topologies are used in NoC at the based on the application. It has been affected the fundamental parameters like area, latency and throughput. There has immense work done in the topology of NoC. A lot of research work done at different topologies for superior results direct topologies are Mesh, torus, ring, star, spidergon, octagon etc[6] and indirect topologies are tree/fat tree and matrix etc. In some NoC architecture has been using 2D and 3D topologies. In NoC, the mesh topology has popularly using as it provides more parallelism and scalability when compared to other topologies. Mesh topology comes under a variety of application like image processing and security systems. Torus topology is the latest version of a typical mesh topology and the torus topology as invariable as Mesh but head nodes connected to tail nodes in complete directions. A torus provides better path diversity than mesh and routing also minimal. The disadvantage of torus has long wrap around the channel which increased the delay. This had been come out of the folded torus topology. The binary tree is connected like the leaves of a tree. Each router has 4 ports. The number of levels is depending on the number of nodes and if the number of nodes is ‘n’ then the number of levels is log4 n. At the first level of the binary tree, nodes are connected to n/4 to the routers. The binary tree topology has vastly used in DNS system.

4.3 SWITCHING MODE

Switching indicates by which the transmission arrangements (bandwidth, buffer capacity etc.) are allocated to users to provide them. Switching systems have been decreasing the network costs by reducing the number of transmission links required to enable the population of users to communicate. NoC has used two types of switching techniques i.e. circuit switching and packet switching. In circuit switching (CS), the data transfer through a dedicated channel. The channel is set to the resource until data transfer and this channel is reserved until the completion of data transfer. In this way, CS provides guaranteed throughput (GT) and quality of service (QoS). In CS, packets are transferred in a pipelined manner along the channel, and it needs one register to buffering. CS is the most efficient for high network traffic and high transmission rates and another advantage is the bandwidth not changed until data transfer ends. But CS is lack of latency requirement and efficiently resource utilization. Mostly CS has fixed structure so it provides limited flexibility. CS has popularly used in the telephone networks.

To get rid problems of CS, Packet Switching (PS) has been transferred the data into packets[14]. PS has most commonly used for NoC as it provides high bandwidth and efficiently resource utilization. PS has a few limitations because of buffer size and network size. Due to these limitations, PS has given the low saturation point and channel latency in case of network size is high. In PS, data have broken into number of packets and further into flits (flow control digits) and further divided into number phits(physical units). PS shows how data information had broken into packets again into flits. The flits are consists of 3 parts of packet i.e. head, body and tail flit. The head flit consists of source, destination and routing information and body flit has the original data information. Lastly, tail flit has consists of information about the end of data packets and upon receiving the tail flit, the router has to release the communication channel.

4.4 CHANNEL ROUTING ALGORITHM

The routing algorithm selects how the data is transferring from source to destination. The routing algorithm has been preventing from deadlock, livelock and starvation. To avoid these overheads, number of routing algorithms has proposed. The routing algorithms had grouped by depending on various routing parameters.

The routing algorithms give important concepts such as 1. Depend on the number of destinations (unicast, multicast and broadcast). 2. Depend on adapting (deterministic, oblivious and adaptive). 3. Depend on routing decision (source, distributive and centralized). 4. Depend on implementation (lookup table and a finite state machine).

Among different types of routing algorithm, adaptive routing algorithm has given low latency results. Deterministic algorithm, simple and inexpensive but Adaptive algorithms are although in theory superior, are complex and power hungry. The deterministic algorithm uses the co-ordinates of X and Y. First, it has to reach X coordinate of destination then it will moves Y coordinate of destination. Consider Xoffset = Xdestination - Xsource and Yoffset = Ydestination - Ysource. If Xoffset = 0 and Yoffset = 0 then the current router is the destination router. If Xoffset<0 then data can be transferred to left side of the current router. If Xoffset>0 then data can be transferred to the right side of the current router and repeat the same process for finding of y co-ordinate. Number of various algorithms proposed in the deterministic algorithm. The second one is the minimal adaptive routing uses the shortest path between the routers. This routing algorithm checks shortest path in every router among available routers. Researchers are having considerable scope of implementing best routing algorithm.

4.5 ARBITER

Whenever number input and output ports are available, selecting the suitable input and output port is complex to the processor. Arbiter has selects the suitable port and different types of arbiters has introduced (fixed priority, round robin fashion) based on the performance. Fixed priority arbiter, it is the simplest arbiter. Priorities are given to the resources and according to their priority, input and output selected. It has easy to implement but the path delay has proportional to the number of input ports. The round robin method gives the highest degree of fairness compared to other arbiters. It will give the priorities to the input and output ports in round robin fashion which will increase the area and latency. To reduce this, distributive round robin method has been used. In this method, input and output selected distributive manner and various types of distributive round robin arbiter comes forward in NoC.

4.6 CROSS BAR SWITCH

The cross bar is used for switching data input port to output port based on Arbiter module. High speed routers are used for the cross bar with full connectivity where at low speed routers are used cross bar without full connectivity. The typical cross bar design has simple when using multiplexer and de-multiplexer. It consists of 'M x N' transmission lines where 'M' horizontal of 'X' axis and 'N' vertical lines of 'Y' axis. The input of the cross bar is given by output of the arbiter. The main problem using cross bar is speed up for transfer of input to output. Parthapratimpande has given the parameters of performance in NoC.

4.7 Distributed Round-Robin

In order to realize arbitration of cyclically distributed incoming requests, the InSIF implements a per-output lane distributed round-robin arbiter (referred to as a distributed arbiter henceforth) to choose a single request from multiple incoming requests for connection to an output lane. Figure 5. shows the distributed arbiter for an output lane. Each distributed arbiter consists of two components: a multiplexer and a counter. To select an incoming request, the counter's output is attached to the MUX's select lines and to control the counter, the MUX's output is attached to the counter's active low enable signal. To exemplify the distributed round robin arbitration process, we consider the initial state where the MUX's inputs/output and counter's output are de-asserted.

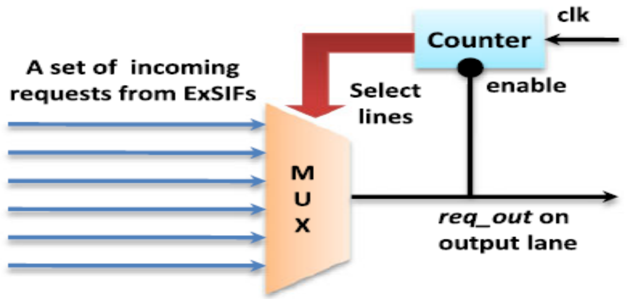

Figure 5: Distributed round-robin arbiter architecture for one output lane.

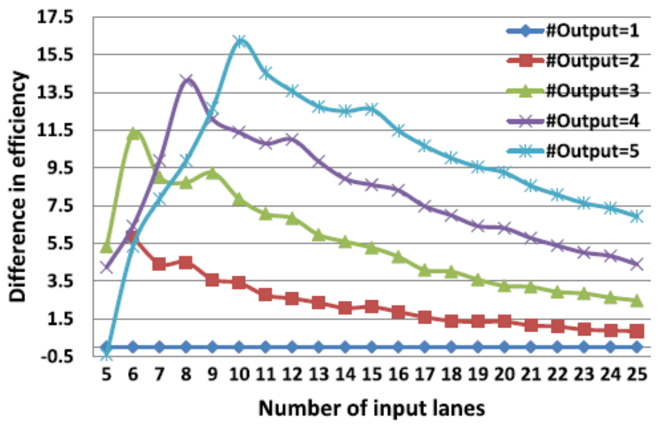

Figure 6: The percetage efficiency difference between the typical and distrubuted round robin arbitters

As the MUX's output is de-asserted in the initial state, the counter is enabled and starts selecting one incoming request per clock cycle (shown as clk in Figure. 6.). As soon as the counter selects an asserted incoming request, the MUX's output is also asserted which, in turn, disables the counter. Disabling the counter holds the current counter value and MUX's select lines persistent thereby keeping a persistent connection between the incoming request and the output lane. De-assertion of the currently selected incoming request de-asserts the MUX's output and enables the counter for selection of subsequent incoming requests. The distributed arbiters' MUX's size and the counter threshold depend on the ratio of the total number of incoming requests and the total number of output lanes (e.g., for a distributed arbiter in the right switch port's InSIF, the counter threshold is ceil ((Kll + Krl + Kd + Kr + Ku)/Kr)). Limiting the counter size limits the service latency (the number of clock cycles before selection of an asserted incoming request signal by the MUX) experienced by an incoming request. As service latency accrues over the channel path, which in turn increases the channel setup latency, the distributed arbiters ensure lower channel setup latency as compared to typical arbiters. Finally, due to a small (only two components) and simple design, distributed arbiters do not adversely affect the switch's area requirements or maximum attainable frequency.

4.8 State Diagram

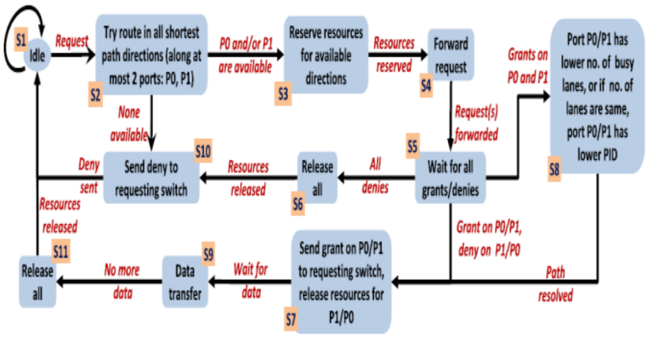
S1
Idle
Request
Try route in all shortest path directions (along at most 2 ports: P0, P1)
S2
P0 and/or P1 are available
Reserve resources for available directions
S3
Resources reserved
Forward request
S4
Grants on P0 and P1
Port P0/P1 has lower no. of busy lanes, or if no. of lanes are same, port P0/P1 has lower PID
S8
None available
Request(s) forwarded
S10
S5
Deny sent
Send deny to requesting switch
Resources released
Release all
S6
All denies
Wait for all grants/denies
Resources released
S11
Grant on P0/P1, deny on P1/P0
Release all
No more data
Data transfer
S9
Wait for data
Send grant on P0/P1 to requesting switch, release resources for P1/P0
S7
Path resolved

Figure 7: State diagram for a switch's channel phase actions and transitions

4.8 Communication Channel Routing

Establishing an arbitrary inter-PE communication channel on a 2D mesh is a straightforward process, however, selecting the best communication channel given all potential routes is challenging. For example, minimal adaptive routing selects the shortest path between two points in a 2D mesh, thus defining the best route as simply the shortest path. However, between two points in a 2D mesh, there are (ΔX+ΔY)!/((ΔX)!*(ΔY)!) equal length shortest path routes where ΔX and ΔY are the differences between the X and Y coordinates of the two points, respectively. Routing algorithms that arbitrarily and/or deterministically choose a particular path can lead to communication bottlenecks and communication resource starvation (all port's lanes are occupied). Our implementation of minimal adaptive routing and the distributed arbiter to resolve the best routing path are unique and minimalistic, and thus afford low area overhead and small latency. We define the best routing path as both a shortest path (with available communication lanes) and a path that best distributes communication channels to avoid communication bottlenecks and communication resource starvation. MACS achieve these routing goals using a minimal adaptive routing state machine.

4.9 Routing State Diagram

Our communication channel routing algorithm is based on minimal adaptive routing to establish, maintain, use (transfer data), and release inter-PE communication channels. These processes correspond to four channel phases: the request service phase, the grant/deny phase, the data transfer phase, and the resource release phase. Each switch can maintain multiple channels, each of which may be in any one of these phases (regardless of the other channel's phases). In the request service phase, a switch identifies destinations for all incoming requests, forwards the incoming requests to neighboring switches towards the destination, and waits for grants/denies. In the grant/deny phase, a switch port receives all grants/denies, resolves between contending grants, if any, based on the number of busy channels and forwards the grants/denies. A communication channel is established at the end of the grant/deny phase if the grant/deny phase received at least one grant from the neighbouring switch (es). In the data transfer phase, the switch maintains the channel based on several status registers, populated by the request service phase and thegrant/deny phase, and transfers data through the communication channel. The end of data transfer triggers the resource release phase. In the resource release phase, the communication channel is destroyed and all resources, such as the status registers related to the communication channel, are reset depicts a state diagram of channel phase actions and transitions. The switch begins operation in the idle state (S1). If the switch receives a channel establishment request and associated channel establishment information on the req_in and data_in signals, the switch transitions to S2 and begins the request service phase. In S2, the switch compares its address with the destination switch address and determines the potential ports ('P0' and/or 'P1' in S2) in which to forward the incoming request (according to the minimal adaptive algorithm). If the current and destination switch addresses do not match, the requested PE is not connected to the current switch and the current switch must forward the request to the switch's neighbouring switches. If the current and destination switch addresses match, the request is forwarded to the appropriate local PE port. If there are available output port lanes, the switch transitions to S3 to establish input-output lane connections and complete the request service phase. In S3, the switch adds the appropriate entries to the status registers. After adding these entries, the switch transitions to S4 and forwards the request to the available lane on port(s) 'P0' and/or 'P1'. Alternatively, if there are no available output port lanes, the routing fails and the switch transitions to S10, sends a deny response to the requesting switch, and transitions back to the idle state (S1). After request service phase completion (i.e., after request forwarding in S4), the switch enters the grant/deny phase (S5) and waits for the grant and/or deny signal responses (gnt_in or dny_in, respectively) on the output lane of port(s) 'P0' and/or 'P1'. If only one port (direction 'P0') was selected in the request service phase and a grant is received, the switch transitions to S7 and forwards the gnt_in, dny_in, and ful_in signals to the corresponding gnt_out, dny_out, and ful_out signals of the requesting port's input lane (the port's input lane in which the request generated from). If two ports (both directions) were selected in the request service phase, there are three possible state transition situations. In the first situation, the switch receives denies from both ports and transitions to S6 to release all channel resources associated with both ports. The switch transitions to S10, sends a deny to the requesting port's output lane and transitions back to the idle state (S1). In the second situation, the switch receives one grant (e.g., associated with 'P0') and one deny (e.g., associated with 'P1'). The switch transitions to S7, forwards gnt_in, dny_in, and ful_in from 'P0' to the requesting port's input lane, and releases the resources associated with ‘P1’ (note that the case is similar when ‘P0’ receives a deny and ‘P1’ receives a grant.) In the third scenario, the switch receives grants from both ports and transitions to S8 for grant resolution. Grant resolution selects the best channel to establish by evaluating the port responses and associated route costs to determine which response to forward to the requesting port’s input lane. Route cost is defined as the number of lanes already assigned to existing communication channels. If both ports route costs are different, the switch selects the lowest route cost port as the best port. If both ports route costs are equal, the switch selects the port with the lowest PID as the best port. Due to this lowest route cost-based port resolution and best port selection, MACS distributes new communication channels towards less congested portions of the network, there by achieving communication load balancing. After the best port is selected (arbitrarily denoted as ‘P0’ in S8 and S7), the switch transitions to S7, forwards gnt_in, dny_in, and ful_in of port ‘P0’ to the requesting port’s input lane, and releases the resources associated with port ‘P1’. This algorithm is deadlock free because in all situations, the switch forwards/ sends either a grant or a deny to the requesting input port, which prohibits infinite channel locking. The total number of cycles required for releasing all of the resources is linear with ($\Delta X + \Delta Y$).

The request service phase propagates successively down all shortest paths from the source switch to the destination switch simultaneously. The grant/deny phase propagates successively backward on all of these paths. Even though multiple request service phase paths may reach the destination switch, only one path will propagate the grant signal all the way back to the source switch, allocating channel resources on this backward propagation. At each switch, if sufficient channel resources exist and that switch is allocated to the routing path (i.e., lies on the best routing path), grant/deny phase completion (S7) and channel establishment occur simultaneously and pipelined data transfers (S9) can begin along the channel.

The comparison summary of proposed work with previous publication is shown in Table 1.

Table 1. Comparison summary

Sl.No	NoC: Device	Slices	LUTs	FFs	Frequency (MHz)
1	Carara NoC(VC)[7]: Virtex2	**861**	**1722**	**455**	**_**
2	Carara NoC(RC)[7]: Virtex2	**758**	**1515**	**398**	**_**
3	MACS(per-PE): Virtex2	**433**	**785**	**274**	**106.7MHz**
4	Lusala NoC[21]: Stratix3	**_**	**1927**	**1980**	**179MHz**
5	MACS(per-PE): Virtex5	**396**	**780**	**360**	**148.1MHz**
6	MACS(per-PE): Virtex6	**440**	**330**	**439**	**152.09MHz**

Data transfer pauses/resumes when the ‘full’ signal (associated with the destination PE’s temporary inability to accept more data) is asserted/de-asserted by the destination switch (denoted by the S9 to S11 and S11 to S9 transitions, respectively). Since MACS uses a circuit switching methodology, in-order data arrival is guaranteed. The channel remains established until the switch enters the resource release phase (i.e., there is no more data to transfer) to free all associated channel resources. The switch transitions to S12 and releases channel resources by removing the corresponding status register entries. The switch enters the resource release phase (S12 and S6) under two situations. The first situation occurs when a PE requests a data transaction termination (S12). In the second situation, the resource release phase occurs simultaneously with the grant/deny service phase when a communication channel establishment request is denied (S6) (there are insufficient channel resources or the switch does not lie on the best routing path). Communication channel establishment denials can be either internal (a switch generates an internal deny due to grant/ deny resolution) or external (a switch receives a denial from a neighbouring switch).

IV. RESULT

The reported implementation results of the proposed NoC BEDT are for effective communication. The results are paid according to the topology, type of switching and routing algorithm. The topology mainly indicates the area of flit width and buffer size. The most common flit width is 32 bits which very lesser than 256 bits. Some proposals used flit width is 20 and 128. Note that the flits are used for flow control, but the bits are used as the number of parallel wires between the routers. However, they may have the exact same width in most cases. Buffers are generally constructed by the flip flops. A lot of proposals report that buffers will take 50-90% of the router area which leads to more power consumption. Virtual channels of each port are associated with the buffers. These are reducing the blocking conditions and improve the performance. Virtual channels are required in adaptive routing algorithms to avoid the deadlock problem. The comparison on summary of proposed work as shown in Table 1.

The simulation results of a proposed single NoC switch, it consists of four directions along with left and right directions and the request command to select required direction. The request would be in the form of 2^n, where n is a number bit. The following commands are used for request selection:

If n=0 then 2^n is 1 and it is for left direction
If n=1 then 2^n is 2 and it is for right direction
If n=2 then 2^n is 4 and it is for east direction
If n=3 then 2^n is 8 and it is for west direction
If n=4 then 2^n is 16 and it is for north direction
If n=5 then 2^n is 32 and it is for south direction

The simulation results for the numbers 1,2,4,8,16 and 32 are in decimal numbers or the numbers 1, 2, 4, 8, 10 and 20 are in hexadecimal numbers. The simulations of top module result are shown in Figure 8.

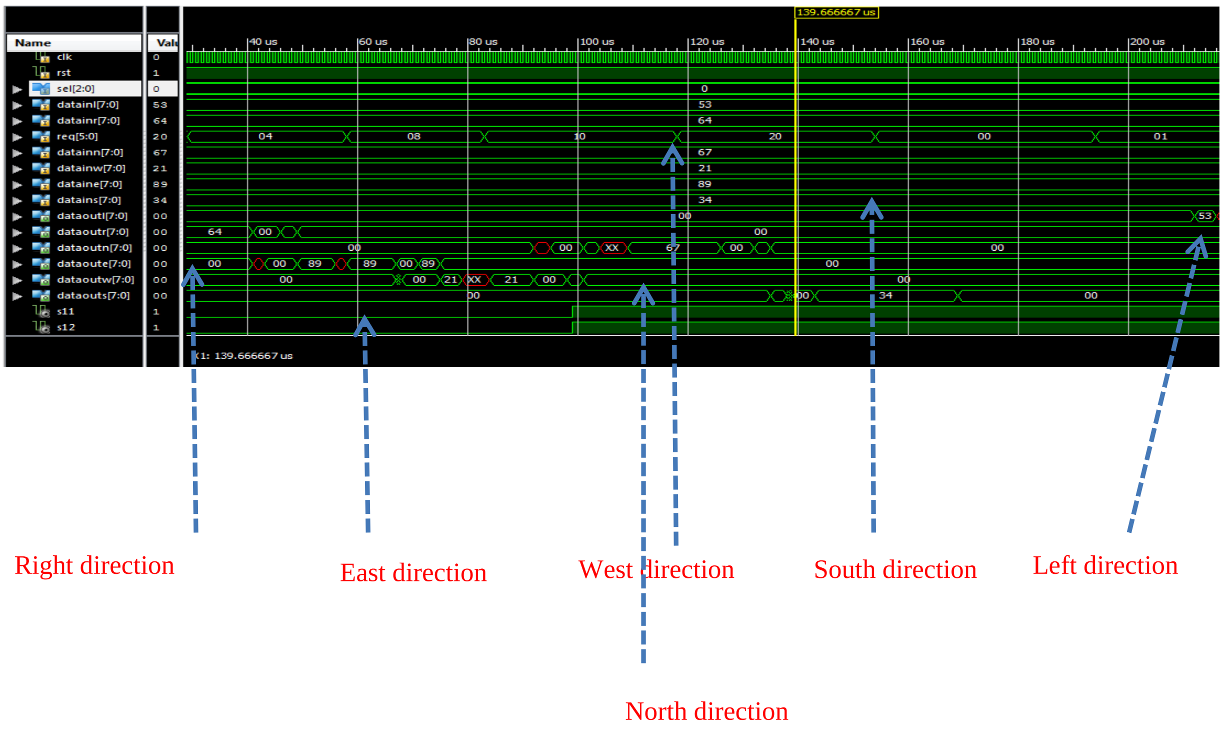

Figure 8: Simulation result of top-module

V. CONCLUSION

In this paper, we have implemented MACS architecture which uses a minimal adaptive routing algorithm with multiple path evaluation and fair path resolution to maximize bandwidth utilization.

The complete architecture has been implemented on FPGA board and which consumes less area, power and with high frequency.

V. REFERENCES

[1]. Wiklund-liu, switched network on chip for hard real time embedded systems, IEEE Computer Society, 2003, p.8.

[2]. Bobda-ali, Dynamic interconnection of reconfigurable modules on reconfigurable devices, IEEE Des. Test comput., Vol.22, No.5, pp.443-451, 2005.

[3]. E Carara, N Calazans, F Mores, A new router architecture for high-performance intrachip networks, J. Integrated Circuits Syst., Vol.3, No.1, pp.23–31, 2008.

[4]. A. Jara-Berrocal and A. Gordon-Ross, SCORES: A scalable and parametric streams-based communication architecture for modular reconfigurable systems, in Proc. Des., Autom. Test Eur. Conf., 2009, pp. 268–273.

[5]. J. Lin and X. Lin, Express circuit switching: Improving the performance of bufferless networks on-chip, in Proc. IEEE First Int. Conf. Network Comput., Nov. 2010, pp. 162–166.

[6]. AK Lusala and JD Legat, A sdm-tdm based circuit-switched router for on-chip networks, in Proc. Reconfigurable Commun.- centric Systems-on-Chip 6th Int. Workshop, Jun. 2011, pp. 1–8.

[7]. N Teimouri, M Modarressi, Power and performance efficient partial circuits in packet-switched networks-on-chip, in Proc. IEEE 21st Euromicro Int. Conf. Parallel, Distrib. Netw. Process, Feb. 2013, pp. 509–513.

[8]. Marta Ortín-Obón, DaríoSuárez-Gracia, Analysis of network-on-chip topologies for cost-efficient chip multiprocessors, microprocessors and Microsystems,5 feb 2016,pp:1-13.

[9]. Rohith Kumar, Gordon Ross, MACS: A highly customizable low-latency communication architecture, Vol.27, No.1, 2016.