Research Journal of Applied Sciences 9 (7): 389-396, 2014
ISSN: 1815-932X

An Efficient Multi Port Network on Chip Router Architecture for Reliable Networks

[1]R. Anitha and [2]P. Renuga
[1]Peri Institute of Technology, Kanchipuram, Tamil Nadu, India
[2]TCE, Madurai, Tamil Nadu, India

Abstract: In this research, researchers propose a memory-efficient on-chip network architecture based router design. In addition to the resource utilization of router design, the high requirements of memories in router architecture design increases the network latency. To overcome such drawback, researchers have developed an optimized weighted scheduling methodology for router architecture and integrated it in the NI such that the network and memory latencies are significantly reduced. This proposed reliable network-on-chip router can reduce faults in both the router components and the reliability components in real time environment. The area overhead is also reduced by resource multiplexing due to the on-demand buffer assignment at each output port. The proposed work results average network latency(16%) and average memory utilization (22%).

Key words: Routers, network interface, routing node, Routing algorithm, Network-on-Chip (NoC), Adaptive Network-on-Chip (AdNoC), on-chip interconnection networks

INTRODUCTION

Nowadays, as the number of processors on a single chip and the computing complexity increasing, the interconnection and communication mechanism among the processors become important factors affecting the performance of chip-multiprocessor. It needs more effective communication and interconnection among the processors to improve performance rather than relies on their processing speed. It needs full consideration of communication demands and characteristics of all kinds of processors and should provide better data transmission performance in limited conditions. Such as chip area, power consumption, data bandwidth and so on. Therefore, it requires higher demands for on-chip communication such as high speed, high throughput, high bandwidth while small area and low power consumption. Traditional interconnection architecture of the chip-multiprocessor such as on-chip bus, crossbar and so on can't satisfy these requirements because of the problems of reusability, flexibility and scalability. It needs a more perfect and effective interconnection technology. NoC has been a research hot spot because of its mass data processing, multitasking parallel computing, scalability and flexibility. It uses routing and packet switching technology, adapts the message communication model and connects resources to network communication. This reduces chip area and power consumption and improves system performance, reliability, reusability and scalability. It adapts the communication requirement for "high data throughout and low communication delay" in chip-multiprocessor. At the same time, the communications among the processors depend on routing nodes with short link which can effectively resolve interconnection delay. NoC uses for layered idea of communication protocol which provides feasibility to overall control power consumption from physical level to system level. NoC architecture has a trend to replace on-chip bus architecture and appears as a better effective architecture for on-chip communication in large scale complex chip-multiprocessor in future. The 2×4 NoC interconnection architecture is mainly used and it is a heterogeneous structure, main processor as a control-intensive processor for system management and controlling while eight DSPs as compute-intensive processors for mass data processing. It uses on-chip bus and NoC interconnection architecture, on-chip bus connects main processor, shared memory and external devices while eight DSPs interconnect each other with NoC interconnection architecture which communicates with main processor through on chip bus.

In this study, researchers propose a memory-efficient on-chip network architecture based router design. In addition to the resource utilization of router design, the high requirements of memories in router architecture design increases the network latency. To overcome such drawback, researchers have developed an optimized weighted scheduling methodology for router architecture and integrated it in the NI such that the network and memory latencies are significantly reduced.

Corresponding Author: R. Anitha, Peri Institute of Technology, Kanchipuram, Tamil Nadu, India

Literature review Lee *et al.* (2006) proposed an energy-efficient Network-on-Chip (NoC) for possible application to high-performance System-on-Chip (SoC) design. It incorporates heterogeneous Intellectual Properties (IPs) such as multiple RISCs and SRAMs, a reconfigurable logic array, an off-chip gateway and a 1.6 GHz Phase-Locked Loop (PLL). Its hierarchically star connected on-chip network provides the integrated IPs which operate at different clock frequencies with packet-switched serial-communication infrastructure. Various low-power techniques such as low-swing signaling, partially activated crossbar, serial link coding and clock frequency scaling are devised and applied to achieve the power-efficient on-chip communications. The 5×5 mm^2 chip containing all the above features is fabricated by 0.18 μm CMOS process and successfully measured and demonstrated on a system evaluation board where multimedia applications run. The fabricated chip can deliver 11.2 Gbp aggregated bandwidth at 1.6 GHz signaling frequency. The chip consumes 160 mW and the on-chip network dissipates <51 mW.

DeOrio *et al.* (2012) present a fault-tolerant architecture, Vicis and companion routing protocol that is robust to a large number of permanent failures, allowing communication to continue in the face of permanent transistor failures. Vicis makes use of a two-level approach. First, it attempts to work around errors within a router by leveraging reconfigurable architectural components. Second, when faults within a router disable a link's connectivity or even an entire router, Vicis reroutes around the faulty node or link with a novel, distributed routing algorithm for meshes and tori. Tolerating permanent faults in both the router components and the reliability hardware itself, Vicis enables graceful performance degradation of networks-on-chip.

Anjum *et al.* (2011), a tool named as CINSIM (Component Based Interconnection Network Simulator) was designed for routing in Linux Systems. The performance evaluation of different combinations of routing algorithms, switching techniques, simulation types, traffic sources and measurement types were done. Al Faruque *et al.* (2012) presented an approach to overcome the problems due to user behaviour or varying workloads with a runtime Adaptive Network-on-Chip (AdNoC). They presented an Adaptive Route Allocation algorithm which provides a required level of QoS (guaranteed bandwidth) coupled with an adaptive buffer assignment scheme which reassigns buffer blocks on-demand. Furthermore, the adaptivity requires a comprehensive, hardly intrusive, runtime observability infrastructure, i.e., using monitoring components in order to gather data on the system state. The area overhead introduced by the adaptive scheme can be traded off against the flexibility gained. Moreover, the area overhead is also reduced by resource multiplexing due to the on-demand buffer assignment at each output port (researchers achieved on an average 42% buffer saving in the experiments). Researchers demonstrate the advantage by using various digital media applications and compare our approach to the state of the art static NoC architectures, e.g., Xpipe, QNoC and Ethereal.

Daneshtalab *et al.* (2012) presented a memory efficient on-chip network architecture to handle the latency and resource utilization problems. Each node of the network is equipped with a novel network interface to deal with out of order delivery and a priority-based router to decrease the network latency. The proposed network interface exploits a streamlined reordering mechanism to handle the in-order delivery and utilizes the advance extensible interface transaction-based protocol to maintain compatibility with existing intellectual property cores. To improve the memory utilization and reduce the memory latency, an optimized memory controller is integrated in the presented NI. Experimental results with synthetic test cases demonstrate that the proposed on-chip network architecture provides significant improvements in average network latency (16%), average memory access latency (19%) and average memory utilization (22%).

Kale and Gaikwad (2011) and Sivakamasundari *et al.* (2011), the design of on-chip routers using FPGA based system based on optimizing power consumption and chip area has been proposed. NoC was adopted in this design as the core bus architecture across different spectrum of SOCs.

Ju and Yang (2011) and Jena *et al.* (2011), design of NoC interconnection architecture for eight compute-intensive processors has been presented. By Jena *et al.* (2011), researchers proposed a PSO based integrated design space exploration framework for the NoC design. Their design space exploration framework has two important steps: mapping and analytical modeling to reduce the energy consumption.

The NoC design for multimedia applications was proposed by Choudhary *et al.* (2011). The method uses a priori knowledge of the applications, communication characteristic to generate an optimized network topology along with chosen routing function compliant routing tables to improve communication performance while improving traffic load distribution. A test wrapper was proposed for routers to test them online and structurally (Babaei *et al.*, 2011). The method made use of Test Access Mechanism (TAM) to transfer the test data from test generator to core-under test.

MATERIALS AND METHODS

The block diagram for the proposed network interface architecture is shown in Fig. 1. The modules present in the router architecture are explained in the study.

Hybrid NI architecture: The NI is divided into the master NI and slave NI. Both NIs are partitioned into two paths: forward and reverse. The forward path transmits the AXI transactions received from an IP-core to a router and the reverse path receives the packets from the router and converts them to AXI transactions. The hybrid model is formed by combining the master-side and slave-side NIs. The master and slave NIs are explained as follows.

Master-side NI: The forward path of the master NI transfers the requests to the network and is composed of an AXI queue, a PU and a RU while the reverse paths, receiving the responses from the network is composed by a packet-queue, a DU and the RU. The RU is a shared module between the forward and reverse paths.

AXI-queue: The AXI master transmits write address, write data or read address to the NI through channels. The AXI-queue unit performs the arbitration between write and read transaction channels and stores requests in either write or read request buffer. The request messages are sent to the PU if admitted by the RU and on top of that a SN for each request should be prepared by the RU after the admittance.

Packetizer: It converts incoming messages from the AXI-queue unit into header and data flits and delivers the produced flits to the router. Since, a message is composed of several parts, the data is stored in the data buffer and the rest of the message is loaded in corresponding registers of the header builder unit. After the mapping unit converts the AXI address into a network address by using an address decoder, based on the request information loaded on related registers and the SN provided by the reorder buffer, the header of the packet can be assembled. Afterward, the flit controller wraps up the packet for transmission.

Packet-queue: This unit receives packets from the router and according to the decision of the RU a packet is delivered to the DU or reorder buffer. In fact, when a new packet arrives, the SN and T-ID of the packet are sent to the RU. Based on the decision of the RU, if the packet is out-of order, it is transmitted to the reorder buffer and otherwise it is delivered to the DU directly.

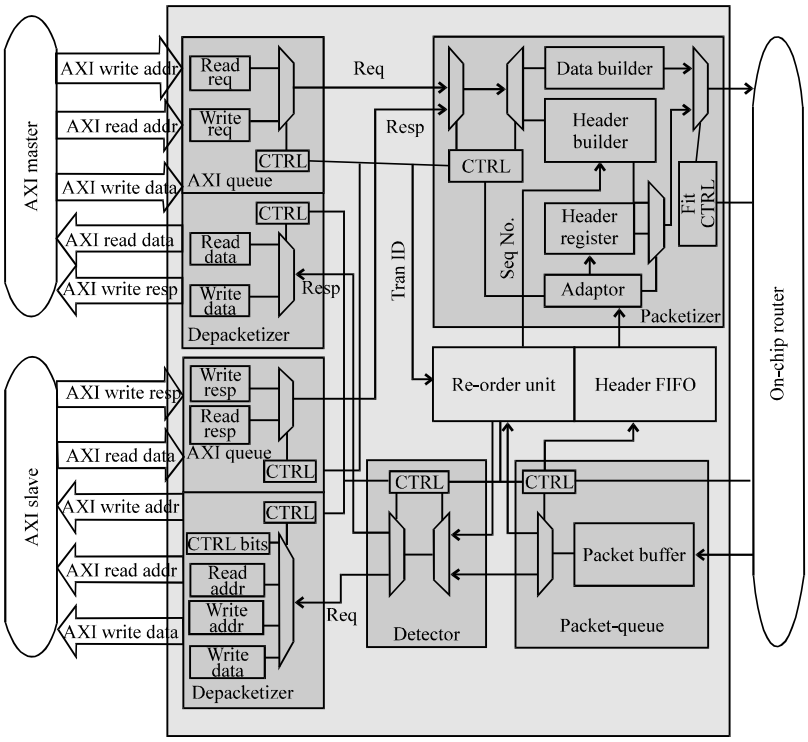

Fig. 1: Proposed NI router architecture

Depacketizer: The main functionality of the DU is to restore packets coming from either the packet-queue unit or reorder buffer into the original data format of the AXI master core.

Reorder unit: It is the most influential part of the NI including a status-table, a reorder-buffer and a reorder-table. In the forward path, preparing the SN for corresponding T-ID and avoiding overflow of the reorder buffer (by an admittance mechanism) are provided by this unit. On the other side, in the reverse path, this unit determines where the outstanding packets from the packet-queue should be transmitted (reorder buffer or depacketizer) and when the packets in the reorder buffer should be released to the DU.

Status-table: The state of outstanding messages is kept in a table named status-table. The status-table has n-entries where each entry corresponds to a T-ID and n is the number of AXI T-IDs. Each entry contains the information of outstanding messages associated with that T-ID and includes number of outstanding Messages (NM), Expecting SN (ES) and LMS fields.

Reorder-table and reorder-buffer: Each row of the reorder-table corresponds to an out of order packet stored in the reorder-buffer. This table includes the valid tag (v), the T-ID, the SN as well as the head Pointer (P).

Slave-side NI: A slave IP-core cannot operate independently. It receives requests from master cores and responds to them. Hence, it is not required to use reordering mechanism in the slave NI. To avoid losing the order of header information (T-ID, SN and so on) carried by arriving requests, a FIFO has been considered. After processing a request in the slave core, the response packet should be created by the packetizer. In order to generate the response packet, the header content of the corresponding request is invoked from the FIFO and some parameters of the header (destination address, packet size and so on) are modified by the adapter. However, the components of slave-side interface in both forward and reverse paths are similar to the master-side interface components except the RU.

Internal architecture of router: The conventional router uses a routing table to determine whether to keep, forward or discard the packets. As the network size increases, the memory requirement of the routing table increases proportionally. Also, the average search time increases as the routing table increases. The node architecture of the proposed router is depicted in Fig. 2. The router

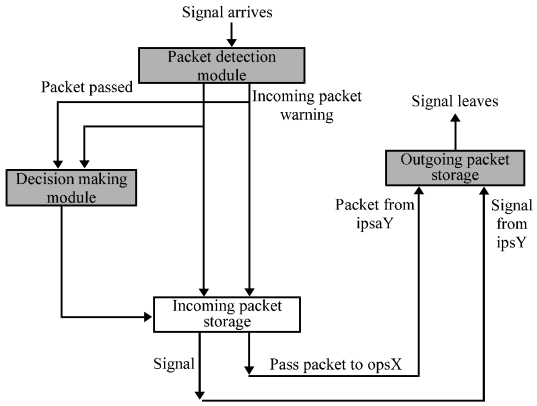

Fig. 2: Node architecture of proposed router

architecture has several internal modules and is explained in the study. Figure 3 illustrates the architecture of multi-port router. The multi-port architecture comprises of virtual channel arbiter, virtual channel buffers, TDM block and input decoding block.

Packet Counter Module (PCM):

- The PCM module strips the destination address from the packet
- It counts the incoming packet bits and sets the RECEIVING-ADDRESS flag when the first bit of the address is read

Address Counter Module (ACM):

- The ACM keeps the track of the number of address bits that have been read
- It indicates which portion of the address is being received (latitude or longitude) and if the entire address is received, it sets the ADDRESS-RECEIVED flag
- When the ADDRESS-RECEIVED flag is set, the IPS is told to keep the packet

Router Address Pipeline (RAP):

- The RAP stores the router address and pipes it the serially so that it can be compared with the incoming destination address
- The router address is static value and is kept in non-volatile memory for long term storage

Address Differentiation Module (ADM):

- The ADM simply compares the ith bit of the destination address to the ith bit of the router address as the destination address is read in to the DMM

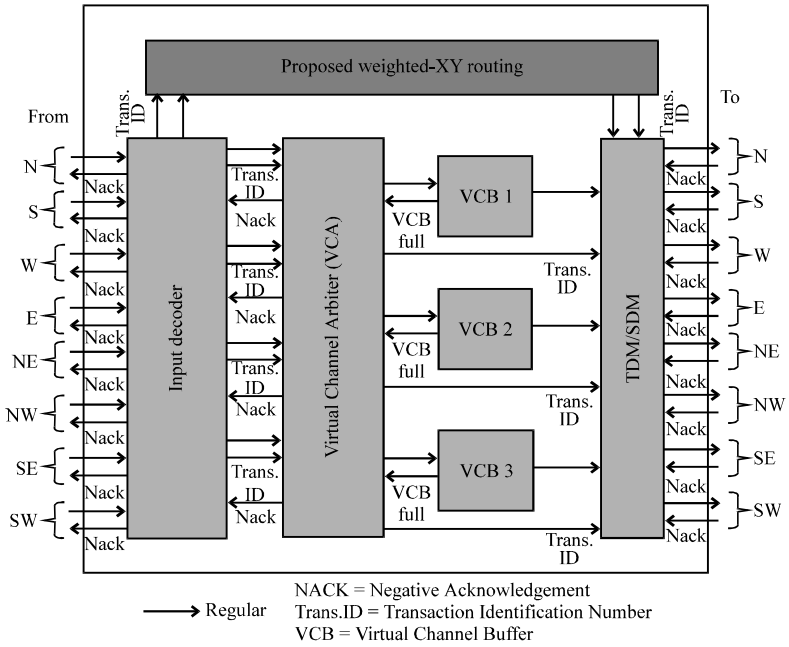

Fig. 3: Architecture of multi port router

- If the DA does not equal to RA, the ADM tells the Incoming Packet Storage (IPS) module to destroy the stored packet

Packet format: Each packet traverse through the router architecture should have the following structure. The total packet size including start and end of packet is 44 bits in size. This structure of packet is explained in the following study.

SOP	M	DLAT	DLONG	SLAT	SLONG	DATA	EOP

SOP → Start of Packet (6 BITS-6'b111111)
EOP → End of Packet (6 BITS-6'b111111)
M → Cast (Unicast-4'b0000, Multicast-4'b0001)
DLAT → Destination Lattitude
DLONG → Destination Longitude

Methodology of router modules:

- All flits that traverse a router first enter the Input Decoder (ID)
- Here, packet information such as a unique transaction identifier (transaction ID), the required connection bandwidth and the destination is extracted from header flits
- This information is sent to the wXY-routing component and the flit is forwarded to the VCA
- The VCA is in charge of sending incoming flits to an appropriate VCB
- If a VCB is full, it notifies the VCA which then stops adding new incoming flits and then turns away any further flits destined for the VCB by sending NACK messages to the previous router
- It must also keep track of and avoid any faulty buffers
- For the output, the flits are removed from the VCBs through a Time Division Multiplexer (TDM)
- A Space Division Multiplexer (SDM) then sends the flits to the corresponding outputs

Virtual Channel Buffer (VCB): The number of Virtual Channel Buffers (VCBs) per port has typically been a design-time parameter. On-demand buffer assignment is employed where VCBs are tied to routers and not to individual ports allows a router to distribute the VCBs as needed to any possible route. Thus, the VCBs are assigned to transactions which are in turn assigned to an output port.

Each VCB holds at most one transaction and as a result only sends flits to one output port. If the adaptive router can find a suitable route for all of the transactions, then the VCBs are assigned according to the transaction direction and ends successfully. Physically, it is realized by using a pool of FIFOs connected to each output port. If the Virtual Channel Arbitrator (VCA) fails to find a free VCB to hold the incoming packet, it is removed from the network. For a requesting transaction, the route is checked in every possible direction.

Modified Weighted Routing algorithm: To provide bandwidth guarantees in AdNoC, the underlying communication infrastructure needs to provide an adaptive route allocation scheme motivated from the adaptive routing schemes for large scale networks. In a

physically static NoC, the routing decision can be distributed or a source-based deterministic routing scheme may be employed. In a distributed deterministic routing scheme, the routing decision is determined locally at each router using predefined rules, e.g., XY-routing algorithm in the QnoC (Bolotin *et al*., 2004) architecture. The source-based deterministic routing scheme (e.g., Xpipe) keeps the complete route in the header of transaction packets and needs the global view of the whole chip before execution or even at design time. That is why both schemes are not suitable for the AdNoC architecture where the subset of tasks and their mapping may change during runtime. Therefore, finding a route for a given NoC and physical mapping of the application is a major challenge.

For a requesting transaction, the route is checked in every possible direction. The modified weighted XY-routing (wXY-routing) algorithm presented in this research, assigns each output port a weight based on available bandwidth and dx, the x coordinate (columns) distance or dy, the y coordinate (rows) distance between the current and the destination node. This ideally gives the packet a maximum number of sensible routing choices along its route as it allows the packet to be routed toward its destination in both the x and y directions. The weight is also proportional to the available bandwidth. If the output port is chosen with the highest associated available bandwidth, the used bandwidth is distributed as evenly as possible among the output ports. Thus, the other output ports are more likely to be able to accommodate future transactions. By allowing both values to contribute to the weight, the weight becomes a trade-off between these two considerations.

The modified wXY routing is described as follows: given is the tuple $\mathcal{T}$ = {N, E, S, W, P}. Each $i \in \mathcal{T}$ has a weight w_i and available bandwidth b_i with $b_i \le b_{max}$ and b_{max} being the maximum link capacity represented as bandwidth. The current router coordinates are x, y. Each packet p has destination coordinates x_d, y_d and a required bandwidth b_p. The weights are assigned using the Eq. 1-8:

$$W_N = \begin{cases} b_N \times |y_d - y| + b_{max} & y_d - y < 0 \\ 0 & b_N < b_p \\ b_N & \text{else} \end{cases} \quad (1)$$

$$W_E = \begin{cases} b_E \times (x_d - x) + b_{max} & x_d - x > 0 \\ 0 & b_E < b_p \\ b_E & \text{else} \end{cases} \quad (2)$$

$$W_S = \begin{cases} b_S \times (y_d - y) + b_{max} & y_d - y > 0 \\ 0 & b_S < b_p \\ b_S & \text{else} \end{cases} \quad (3)$$

$$W_W = \begin{cases} b_W \times |x_d - x| + b_{max}, & x_d - x < 0 \\ 0 & b_W < b_p \\ b_W & \text{else} \end{cases} \quad (4)$$

$$W_{NE} = \begin{cases} (b_N + b_E) \times (y_d - x_d) + b_{max} & y_d - x_d < 0 \\ 0 & (b_N + b_E) < b_p \\ b_N + b_E & \text{else} \end{cases} \quad (5)$$

$$W_{NW} = \begin{cases} (b_N + b_W) \times (y_d - x_d) + b_{max} & y_d + x_d > 0 \\ 0 & (b_N + b_W) > b_p \\ b_N + b_W & \text{else} \end{cases} \quad (6)$$

$$W_{SE} = \begin{cases} (b_S + b_E) \times (y_d - x_d) + b_{max} & y_d - x_d < 0 \\ 0 & (b_S + b_E) < b_p \\ b_S + b_E & \text{else} \end{cases} \quad (7)$$

$$W_{SW} = \begin{cases} (b_S + b_W) \times (y_d + x_d) + b_{max} & y_d + x_d > 0 \\ 0 & (b_S + b_W) > b_p \\ b_S + b_W & \text{else} \end{cases} \quad (8)$$

They are calculated to be proportional to the distance from source to destination and to the available bandwidth if the output direction is facing the destination and proportional to the available bandwidth if it is not. If there is not enough bandwidth available, the weights are zero. The route r chosen is then to the direction with the highest weight:

$$r = \begin{cases} P & x = x_d \text{ and } y = y_d \\ i \in \{N, S, W, E, NE, NW, SE, SW\} & \text{else, } w_i = \max_i(w_i) \end{cases} \quad (9)$$

The wXY-Routing algorithm is designed to make meaningful routing decisions whenever possible. However, in certain worst-case scenarios, routing problems can arise which need to be dealt with. A livelock situation occurs when a packet is routed through a network without ever reaching its destination. However, since the wXY-Routing algorithm routes towards a destination whenever possible, a livelock can only happen if a packet is continuously misrouted away from its destination. This is a sign that there are no available routes to the destination.

Routing methodology for 4 port: While, routing of a packet in 4 port architecture, the address of the packet, i.e., the destination address and the node address are compared. For each comparison, the direction of the destination is chosen. The comparison of addresses and selection of directions are clearly shown in Table 1.

Table 1: Selection of direction in 4-port network

Address	Functions
Dest_lat = node_latDest_long = node_long	Router keeps the packet
Dest_lat>node_latDest_long>node_long	Packet routed to North port
Dest_lat<node_latDest_long>node_long	Packet routed to South port
Dest_lat = node_latDest_long>node_long	Packet routed to East port
Dest_long<node_long	Packet discarded

Table 2: Selection of direction in 8-port network

Address	Functions
Dest lat = node latDest long = node long	Router keeps the packet
Dest lat<node latDest long<node long	Packet routed to Southwest port
Dest lat>node latDest long = node long	Packet routed to North port
Dest lat>node latDest long<node long	Packet routed to Northwest port
Dest lat<node latDest long = node long	Packet routed to South port
Dest lat>node latDest long>node long	Packet routed to Northeast port
Dest lat = node latDest long<node long	Packet routed to West port
Dest lat = node latDest long>node long	Packet routed to East port
Dest lat<node latDest long>node long	Packet routed to Southeast port

Routing methodology for 8 port: Routing of a packet in 8-port architecture, follows the same procedure as in 4-port architecture, i.e., the address of the packet, i.e., the destination address and the node address are compared. For each comparison, the direction of the destination is chosen. The comparison of addresses and selection of directions are clearly shown in Table 2.

RESULTS AND DISCUSSION

To evaluate the performance of the proposed schemes, uniform and non-uniform/localized synthetic traffic patterns are considered separately. In the non-uniform mode, 70% of the traffic is local requests where the destination memory is one hop away from the master core and the remaining 30% of the traffic is uniformly distributed to the non-local memory modules. Researchers also consider the hotspot traffic pattern where four memory nodes are chosen as hotspots receiving an extra portion of the traffic (10%) in addition to the regular uniform traffic. For the uniform and hotspot traffic profiles, researchers obtained very similar performance gains in each configuration, though they are not presented due to the lack of space. For appraising the area overhead of the proposed architectures each scheme is synthesized by Xilinx Project Navigator tool using the 90 nm CMOS technology.

The layout areas and power consumptions of the master-side, slave-side, hybrid interfaces, different memory controller, are listed in Table 3. As can be seen from the Table 3 using the hybrid architecture for the proposed router architecture is more beneficial (in terms of power and area) than using the master-side and slave-side models when each node is composed of a dedicated processor and memory. That is using a hybrid NI model reduces 14.3 and 13.8% in hardware area and power dissipation, respectively. Because all queues (and FIFOs) are equal in size, they do not affect the comparison. On the other hand, the master-side and slave-side NI architectures are more cost efficient if each node consists of a dedicated processor or memory as in the former configuration (A). Also, comparing the area cost of the baseline model to each proposed NI indicates that the hardware overheads of implementing the proposed NI schemes are <0.5%.

Table 3: Utilization of power consumption for NI modules

Components	Area (mm^2)	Power consumption (mW)
Master mode	0.0863	22
Slave mode	0.0623	18
Hybrid	0.1753	16
Memory utilization	0.1461	17

The proposed design also provides an Adaptive Route Allocation algorithm to meet varying bandwidth guarantees and an on-demand buffer block assignment scheme for runtime connection establishment between various ports of router architecture. Furthermore, researchers have compared the proposed router component with the state of the art monitoring component inside the aethereal architecture. The traffic generated by the monitoring component of AdNoC cannot directly be compared to that of the aethereal monitoring component. The occurrence of events in the aethereal monitoring component is different to ours as they are mainly managing events that are necessary for debugging whereas researchers are managing different types of events that are needed to adapt the on-chip communication architecture (Fig. 4).

The monitoring component of the NI architecture is event-based and the event-counter values are stored in a LUT. One entry in the LUT ties the transaction ID to the source of the transaction and to its associated counters. Transaction numbers are added to the LUT when a header flit of the transaction arrives at a router. Once an event occurs it initiates a read from the LUT using the event transaction ID. It then compares the counter value returned from the LUT of the event type corresponding to the event type that arrived. If the counter value has reached its threshold, the NI part of the monitoring component is informed and the counter value is set to zero in the LUT. If not, the incremented value is written to the LUT. The NI part of the monitoring component, upon receiving an event, first compares the transaction source with its own address. If it is not the sender then a packet is sent to the remote sender. The hardware overhead for the modified wXY-routing, on-demand VCB assignment and monitoring components is given in Table 3. The crossbar to implement the on-demand buffer assignment and the small LUT to keep the VCB pointers contribute to

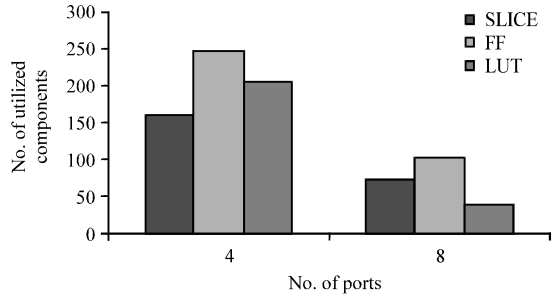

Fig. 4: Performance comparison of proposed work

the area. The implementation of the VCB crossbar for a flit size of 8 bits having four VCBs shared by the four output ports costs 156 slices in an FPGA.

CONCLUSION

In the proposed system, memory-efficient on-chip network architecture based router is designed. In addition to the resource utilization of router design, the high requirements of memories in router architecture design increases the network latency. To overcome such drawback, researchers have developed an optimized weighted scheduling methodology for router architecture and integrated it in the NI such that the network and memory latencies are significantly reduced.

REFERENCES

Al Faruque, M.A., T. Ebi and J. Henkel, 2012. AdNoC: Runtime adaptive network-on-chip architecture. IEEE Trans. Very Large Scale Integr. Syst., 20: 257-269.

Anjum, S., E.U. Munir and M.W. Nisar, 2011. Simulation and performance evaluation of network on chip architectures and algorithms using CINSIM. J. Basic Applied Sci. Res., 1: 1594-1602.

Babaei, S., M. Mansouri, B. Aghaei and A. Khadem-Zadeh, 2011. Online-structural testing of routers in network on chip. World Applied Sci. J., 14: 1374-1383.

Bolotin, E., I. Cidon, R. Ginosar and A. Kolodny, 2004. QNoC: QoS architecture and design process for network on chip. J. Syst. Architecture, 50: 105-128.

Choudhary, N., M.S. Gaur and V. Laxmi, 2011. Routing centric NoC design for high performance multimedia application. Int. J. Soft Comput. Eng., 1: 254-258.

Daneshtalab, M., M. Ebrahimi, P. Liljeberg, J. Plosila and H. Tenhunen, 2012. Memory-efficient on-chip network with adaptive interfaces. IEEE Trans. Comput. Aided Design Integ. Circuits Syst., 31: 146-159.

DeOrio, A., D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, J. Hu and G. Chen, 2012. A reliable routing architecture and algorithm for NoCs. IEEE Trans. Comput. Aided Design Integr. Circuits Syst., 31: 726-739.

Jena, R.K., M.M. Aqel and P.K. Mahanti, 2011. Network-on-chip design space exploration: A PSO based integrated approach. Eur. J. Sci. Res., 64: 5-18.

Ju, X. and L. Yang, 2011. NoC research and practice: Design and implementation of 2x4 2D-torus topology. Int. J. Inform. Technol. Comput. Sci., 4: 50-56.

Kale, M.A. and M.A. Gaikwad, 2011. Design and analysis of on-chip router for network on chip. Int. J. Comput. Trends Technol., 9: 182-186.

Lee, K., S.J. Lee and H.J. Yoo, 2006. Low-power network-on-chip for high-performance SoC design. IEEE Trans. Very Large Scale Integr. Syst., 14: 148-160.

Sivakamasundari, P., R. Sudha and S. Sasikala, 2011. Implementation of an effective router architecture for NoC on FPGA. Proceedings of the International Conference on Advanced Computer Technology, January 2-4, 2011, Bangalore, India.