World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:3, No:3, 2009

Flexible Wormhole-Switched Network-on-chip with Two-Level Priority Data Delivery Service

Faizal A. Samman, Thomas Hollstein and Manfred Glesner

Abstract—A synchronous network-on-chip using wormhole packet switching and supporting guaranteed-completion best-effort with low-priority (LP) and high-priority (HP) wormhole packet delivery service is presented in this paper. Both our proposed LP and HP message services deliver a good quality of service in term of lossless packet completion and in-order message data delivery. However, the LP message service does not guarantee minimal completion bound. The HP packets will absolutely use 100% bandwidth of their reserved links if the HP packets are injected from the source node with maximum injection. Hence, the service are suitable for small size messages (less than hundred bytes). Otherwise the other HP and LP messages, which require also the links, will experience relatively high latency depending on the size of the HP message. The LP packets are routed using a minimal adaptive routing, while the HP packets are routed using a non-minimal adaptive routing algorithm. Therefore, an additional 3-bit field, identifying the packet type, is introduced in their packet headers to classify and to determine the type of service committed to the packet. Our NoC prototypes have been also synthesized using a 180-nm CMOS standard-cell technology to evaluate the cost of implementing the combination of both services.

Keywords—Network-on-Chip, Parallel Pipeline Router Architecture, Wormhole Switching, Two-Level Priority Service.

I. Introduction

Networks-on-Chips (NoC) is a bridge concept from Systems-on-Chip (SoCs) into Multiprocessor System-on-Chip (MPSoC). A SoC design approach uses sometimes more than one processing element (PE) to implement an integrated circuit for a certain system application. The PEs send messages to other PEs for sharing computational processes to complete tasks. A sophisticated communication structure is needed for inter-processor data exchange. Rather than using a bus for single communication among PEs, or using point-to-point communication, a concept of shared segmented communication infrastructures is proposed to support application-scalability.

Research studies with the Arteris NoC [1] have demonstrated the feasibility and advantages of NOCs over traditional bus-based architecture but have not focused on compatible communication standards. Computer bus-based communication architectures do not easily handle the real-time data flows associated with networking, telecommunication, and multimedia data streams [2]. On-chip networks will meet the distinctive challenges of providing functionally correct and reliable operation of interacting system-on-chip components [3].

The effectiveness of NoC platforms for MPSoCs is more and more significant when a huge number of PEs is used in the MPSoC. Therefore, NoCs enable promising concepts for the design of super-computers with multiprocessor cores. NoCs have also potential applications in integrated control systems, e.g. in automotive electronic control and entertainment systems. The NoC concept has potential to provide sustainable platforms and proposes a new paradigm in SoC architecture and multiprocessor systems [4].

F. A. Samman, T. Hollstein and M. Glesner are with Institute of Microelectronic Systems, Darmstadt University of Technology, Karlstr. 15, 64283 Darmstadt, Germany. E-mail: [faizal.samman, thomas.hollstein, glesner]@mes.tu-darmstadt.de.
F. A. Samman is also with Dept. of Electrical Engineering, Hasanuddin University at Makassar. Jl. Perintis Kemerdekaan km. 10 Makassar, Indonesia [E-mail: faizalas@unhas.ac.id]. He is currently pursuing Doctoral Degree with DAAD (*Deutscher Akademischer Austausch-Dienst*) Scholarship at TU Darmstadt, Germany.

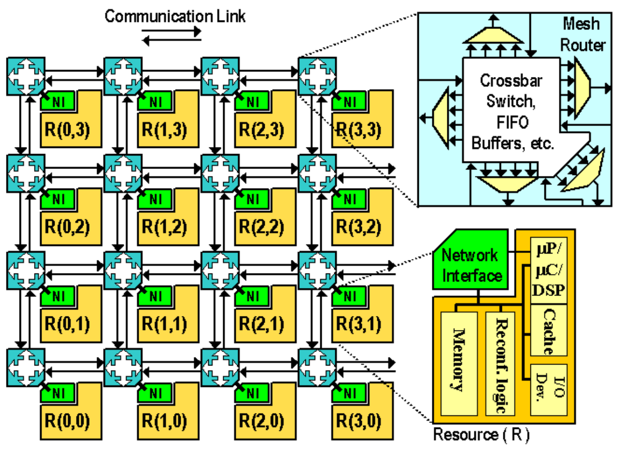


Fig. 1. A 2-dimension mesh 4x4 topology.

Fig. 1 shows an example of a NoC platform with a 4x4 mesh topology. Each mesh router is connected with one resource through a network interface. A resource can be as bus-based platform with one PE (processing element). The PEs can be microprocessors, microcontrollers, or DSPs. Resources can also be represented by RAMs, ROMs, reconfigurable logic blocks, or even an I/O device equipped with ADC or DAC.

The Network topology could influence the scalability and performance of the NoC. The architecture and routing decision must meet bandwidth requirements and should be scalable for wide range of applications. Some NoCs that have been developed with Mesh topology are NOSTRUM [5], SoCBUS [6], RAW [7], PNoC [8], CLICHÉ [9] and HiNoC [10]. OCTAGON NoC [11] uses octagon topology. Fat tree topology is used in SPIN [12], and its extended version DSPIN [13] uses mesh distribution of clusters. Flexible regular and irregular topology is presented in [14]. Xpipes NoC [15] supports a customized topology. ASNoC [16] is an application specific NoC, where the design methodology supports the development of NoCs with 2-D mesh and hierarchical irregular topologies.

Message transmission from a resource to another resource through the intermediate router nodes can be divided into synchronous and asynchronous methods. Asynchronous NoCs are introduced in CHAIN [17], ASPIDA [18], MESCAL [19], PROTEO [20], and ANoC [21], while in MANGO [22] asynchronous clock-less NoC is proposed. In synchronous designs, global clock-trees are distributed, which leads to electromagnetic interference effect and clock power consumption. Asynchronous communication design is a promising concept, but lacks of industrial standard tools support, especially with respect to testability issues. Synchronous NoCs can also support GALS (globally asynchronous, locally synchronous) concept by implementing asynchronous input/output queues in network interfaces. The NoC is considered as a synchronous island, where resources clocked with different frequencies are connected through network interface.

A resource may send a short, medium, long messages, or even a datastream in a certain time duration to other resources. In real-time applications, some messages require lower completion bounds or higher bandwidth. These messages are e.g. run-time reconfiguration data (small until medium size messages), or video stream packets (very large size messages), or message having critical time constraint. Therefore, a NoC providing two priority level for packet services, i.e. for low and high priority messages are introduced in this paper that supports applications running normal traffics and time-critical traffics.

This paper will present a NoC prototype using *Adaptive West-First Routing Algorithm* with two-level priority packet delivery service, i.e. a *low-level priority (LP) packet* and a *high-level priority (HP) packet*, for unicast communication in 2-D mesh topology. The NoC is develop based on synthesizable VHDL modules. Some flexible object modules can be selected and combined with base modules to obtain a specific mesh router prototypes in accordance with a desired specification, starting from standard until advanced NoC models.

The remaining sections of this paper is organized as follows. Section II describes related works and motivation to design our router. The features and characteristic of the router is presented in Section III. Section IV describes the router microarchitecture and the router structure to provide the two-level priority data delivery service. Experimental results by using some traffic scenarios are shown in Section V. Concluding remarks and the future work for potential further investigations are presented finally in Section VI.

II. Related Works and Motivations

Some NoC prototypes have been designed to provide additional services for packets by using virtual channels, as presented by Æthereal [23], NOSTRUM [5], DSPIN [13], MANGO [22] and HiNoC [10]. Æthereal [23], DSPIN [13], MANGO [22] and HiNoC [10] provide two queues to buffer different packets, i.e. Best-Effort Queue and Guaranteed-Throughput Queue. Æthereal [23], DSPIN [13], and HiNoC [10] uses then a time-division multiplexing (TDM) mechanism to switch packets synchronously. While MANGO [22] uses clockless method to switch packets asynchronously.

The implementation of two-level priority for different packet services has been developed in some routers for off-chip network applications. SpiceWire router [24] developed in European Space Agency (ESA) for instance uses an arbitration mechanism including a priority scheme to select which input port is to be served when there are two input ports in a router waiting to use a particular output port. The SpaceWire router has no priority flag available within the header of packet to specify its priority level. The header contains only an address information. Therefore, packet priority must be associated with a logical address or with the input port number. The logical addresses may be assigned high or low priority in a routing tables. The high priority logical addresses have preferential access to an output port when arbitration takes place. A logical address that has been assigned high priority will acts as a high priority channel across the network from many possible sources to the one destination. If high and low priority access to a particular destination is required then two logical addresses are required for a particular destination, one assigned high priority and the other low priority. A source can then decide which logical address to use when sending a packet to a destination, depending on the required priority of the packet.

In the SpiceWire router, it is possible to ensure real-time, deterministic delivery of commands (packets) using priority addresses provided that there are no possible conflicts between packets, i.e. there is only one node sending out priority commands. If deterministic delivery is required, the maximum packet size used on the network must be limited. Alternatively, nodes can take turns to transmit their high priority packets possibly by using time-codes to determine which node transmits next.

Intel Priority Packet [25] provides an utility to set up priority filters to process high priority network traffic before normal traffic. This setup will give priority to time-critical traffic. The Priority Packet filters are used to (1) tag packets with priority levels or forwarding behaviors, (2) drop incoming packets that match certain criteria, and (3) count the number of packets that match certain criteria. By prioritizing traffic at the hosts or entry points of the network, network devices can forward decisions on priority information defined in the packet.

A protocol presented in [26] is defined for mixed data/voice/multimedia communications systems to transmit and receive high-priority, real-time traffic over low-speed digital communication links by embedding such high-priority traffic in low-priority, non real-time traffic. High-priority, real-time packets are thus transmitted without delay by preempting low-priority packets. Low-priority, nonrealtime packets are held during preemption, and low priority transmission is automatically resumed after transmission of high-priority packets has been completed.

A method for processing high priority and low priority packets in a network device is also presented [27]. An arbitration on high priority packets is performed until no high priority packets remain. Afterwards, the arbitration is then enabled on low priority packets. A packet size associated with the selected low priority packet is compared with a programmable threshold. Low priority packets are excluded from subsequent arbitration for a programmable duration when the packet size exceeds the programmable threshold.

The performance of the preemptive priority queueing strategy in an input queueing switch with two priority classes is considered in [28]. The work proposes a new queueing model consisting of two independent input buffers with separate head of line (HOL) queues. This model considers the priority queue structure as well as the influence of the preemptive priority scheme for output contention. By applying the approximation techniques of the flow conservation rule and an equivalent queueing model, the queue length distribution, delay and maximum throughput are obtained in closed-form without using any heuristic adjustments.

The aforementioned implementation of the two-level priority for packet services in the off-chip interconnection networks has motivated us to implemented in our on-chip router. We use also two queues to buffer low and high-priority packets, where preferential access to make routing direction is given to the packet coming from the high priority queue. We use then a locally managed variable packet identity method to organize a wire through-sharing technique that allows flits of different packets are interleaved in the same queue.

III. On-Chip Network Feature, Characteristic

A. Two-Level Priority Message Delivery Service

The early prototype of our NoC has provided only a single priority with *Best-Effort* (BE) service. This paper proposes a new prototype with an additional services for different level of priority. Therefore new modules i.e., a *HP FIFO buffer*, *TypD* unit and *VCSC* are inserted in each port to support the new service as shown in Fig. 3(b).

1) Low Priority Message Service (LP): Our proposed LP service guarantees lossless packet completion and in-order message delivery, but provides no commitment to latency bound or data throughput because the messages are sent with a packet-based approach. The LP packets are routed using a minimal west-first adaptive routing algorithm, where the packets will not be routed away from their target nodes. The routing is made on flit-by-flit basis, where different packets from different input ports share the link wires using wormhole switching, and can be interleaved in the FIFO. The incoming packet flits, which require the same link, are selected by an arbiter unit using a fair round-robin arbitration.

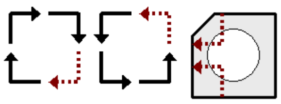


Fig. 2. The turn model of adaptive West-First routing algorithm (the solid lines are allowed turns, and the dashed lines are prohibited turns).

2) Higher Priority Message Service (HP): The HP packets will reserve absolutely the selected links to route the packets into their target nodes. The HP packets are routed using a misrouting or non-minimal west-first routing algorithm, where the packets can be routed away from their target nodes to find an optimal path. The HP packets will firstly be routed into the possible output links, which have not been used by another packet. Non minimal routing will be undertaken as long as the routing does not violate the prohibited turn-models as shown in Fig. 2. Otherwise, the HP packet will select the output link which has more free IDs (see Section 3(a)) and more available free registers in the downstream FIFO. The HP packet will not share the reserved links with other HP packets. These packets will be buffered into the FIFO in virtual HP channels. The arbiter will always prioritize the flits in the HP buffer (higher priority) as a winner to access the output ports. The LP packets must wait until the HP buffer is empty or the last flit of the HP packet has been forwarded from the HP buffer. In this situation, the HP packet uses the full bandwidth of its reserved links.

B. Packet interleaving and Identity-Slot Divison Multiplexing

Contention-free routing has been introduced in Æthereal NoC [23]. The contention-free routing can be implemented by using a time-division multiple access technique. This approach uses a pipelined circuit switching method. Contention in our NoC is handled by using an identity-slot division multiple access technique for wormhole packet interleaving. Each wormhole packet is injected into the network with the same identity-tag (ID-tag). But each time the flits of the packets are forwarded into the next router, their local ID-tag will be updated. Each flit belonging to the same message will have the same local ID-tag in a certain communication link to differentiate it from other flits of other messages (All packets are interleaved in the FIFO buffer). Therefore dynamic packet identity management (IDM) modules are implemented over the link to map old local ID-tag of each flit into new local ID-tag.

C. Congestion-Aware Adaptive Routing

Deadlock is a situation where all packets in the deadlock configuration cannot be forwarded to the next network nodes. Deadlock is formed by a cycle, where there are channel dependencies between packets in that cycle [29]. By introducing one prohibited turn in one turn model, deadlocks can be prevented. Some turn models for west-first, north-last and negative-first for 2-D mesh topology are introduced in [30]. An Extended partially adaptive west-first routing algorithm for a 3-D mesh is proposed in [31].

By introducing a small number of virtual channels [32], [33], deadlock can be avoided. It seems that implementing virtual channel is as good solution for deadlock free router design. However, the use of the virtual channel leads to high area and logic utilization. Design of a deadlock-free routing algorithm that does not use virtual channels is important [34]. This context is also considerable to the NoC hardware implementations.

Our NoC uses an adaptive west-first (WF) routing algorithm, which does not require virtual channels to avoid deadlock configuration. In our NoC, virtual channels are only used in the NoC to provide two priority level for packet service as explained in Section IV-C. As shown in Fig. 2, the turns from North to West and South to West are prohibited. Hence, packets are routed firstly to West when the target nodes are located in North-West or South-West quadrants. Packets can be adaptively routed when destination nodes are located in North-East or South-East quadrants.

Adaptive routers based on neighbor congestion states have been proposed in [35], [32], [36]. Our NoC uses one-hop congestion measurement, i.e. by considering the congestion of the adjacent nodes to select adaptively two possible directions. The adaptiveness of the routing algorithms is also based on the number of free identity-slots provided by identity-slot manager (IDM) modules. The availability of ID-slots in a communication link segment denotes actually that there is still available bandwidth that can be allocated for new incoming messages.

When there are two possible output directions to route a packet, then the router will consider firstly the number of free ID-slot provided by the IDM units at both output ports. Secondly, the router will consider the number of free registers in the FIFO buffer at the two adjacent mesh nodes. The packet will be routed to the output direction, where more free ID-slots are available in the output direction. If the numbers of free ID-slots are the same for two possible directions, then the router will select the direction, where more free registers are available in the next downstream FIFO buffer.

D. Special Packet Format with Extra Control Bits

The packet format used in the NoC is presented in Fig. 3(a)(a). The 38-bit packet consists of a header flit followed by payload flits. Two additional 3-bit heads are Type and ID (Identity) bits. The Type can be a header, a data body, and the end of databody (last flit). The 3-D source and target address of the packet are asserted in the header flit. The Z-address is reserved for further development of 3-D NoC topology. Each message is associated as single packet even if the size of message is very large. It means that each message will have only one header flit for one target node. The message body will travel in the NoC to follow links set up by the header flit, and the end flit of the message will close the link reservation. This approach will guarantee in-order message delivery even if adaptive routing algorithm is used, because the header flit is the only flit, which is routed adaptively to find an optimal link. Each of its flits has the same local identity number (ID-tag) to differentiate it from flits of other packets, when it passes through a communication segment of the NoC. The ID-tag of the data flits of one packet will vary over different communication segments in order to provide a scalable concept.

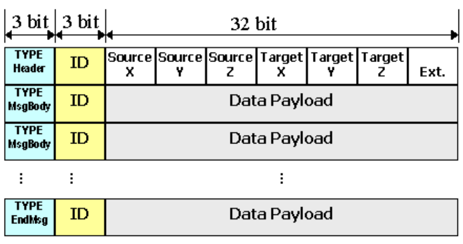
Fig. III-D represents bit encodings for packet types. The LP packet is encoded with binary code ’001’ and the HP packet with binary code ’100’. The remaining binary codes can be used for other types of packet for future investigation, e.g. for connection-oriented guaranteed-throughput packets.

IV. On-Chip Router Architecture

A. Modular Architecture

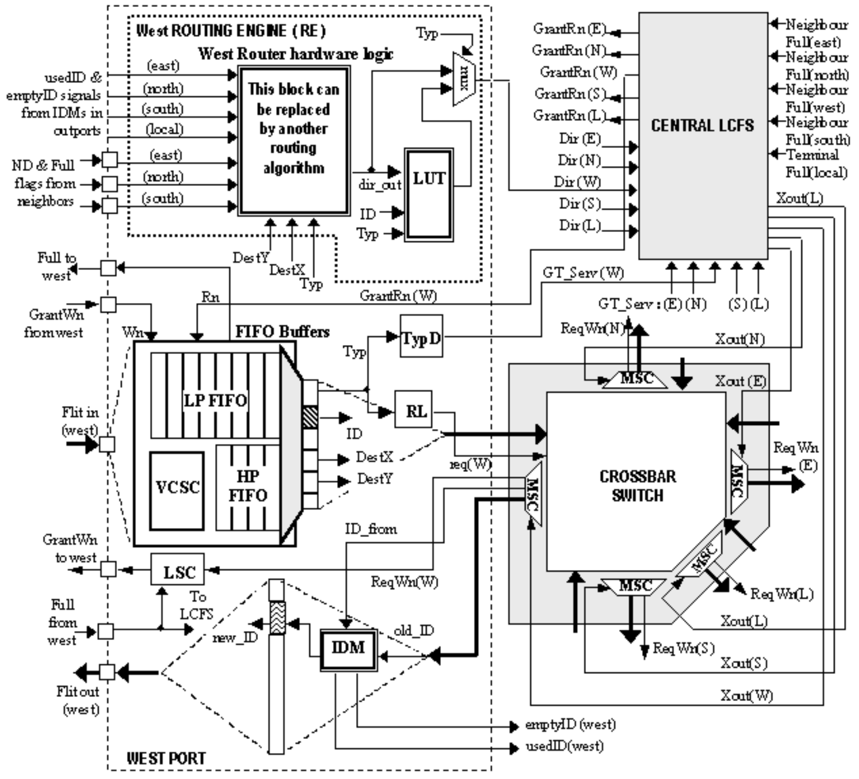
In general, Our NoC consists of modular VHDL objects as shown in Fig. 3(b). For the sake of simplicity, Fig. 3(b) presents only the west port block. A flexible routing engine and parameterizable FIFO buffer modules in all input ports are shown in the block. Arbiter modules for output selection in all output ports are placed in the LCFS (Link Controller and Flow Supervisor) block. This section will only present general architecture of the NoC.

The FIFO buffer sends congestion information, i.e. a full flag and number of data (ND) in the buffer to its neighbor (the west neighbour, as presented in Fig. 3(b)). The full flag is sent to the LCFS in the



Type of Flit	bin/dec code
No FLit	000 (0)
LP Packet Header	001 (1)
HP Packet Header	100 (4)
Message Body	010 (2)
End of Message	011 (3)

(a) Packet format.



(b) Architecture.

Fig. 3. Modular architecture and the special packet format of our NoC with extra 6 control bits (Flit type and ID-tag fields).

west neighbor to control the flit overflow. If the FIFO is full, then the LCFS in the west neighbor will not grant its FIFO to forward any flit to the FIFO. The LCFS in each mesh node receives full flags from all neighbor FIFOs, and sends grant-read signals to all the FIFO in the mesh node to hold or release the flit from the FIFO. The router hardware logic receives signals from neighbor FIFOs and IDMs, and uses these signals to decide the best routing direction for a packet.

B. Centralized Link Controller and Flow Supervisor (LCFS)

The LCFS functionalities are to control a link in a crossbar switch and to supervise neighbour congestion states. The structure of the LCFS is depicted in Fig. 4. It consists of direction assignment unit, decoders, arbiters, winner flit encoder (*EncWin*), multiplexor (*Mux1*) and request filter unit (ReqFilter). The *ReqFilter* unit is used to control the priority of HP packet over LP packets. When HP and LP packets are detected in input ports, then the arbiter will prioritize this packet.

The number of each component follows the number of outports. The LCFS receives routing direction requests from all routing engines, and a full flag from the network interface and the neighbor nodes. The full flags are sent to the arbiters and the direction requests are sent to the decoder.

The decoder unit decodes 3-bit routing direction request signals (EAST, NORTH, WEST, SOUTH, LOCAL) from all input ports into 1-bit signals. And then the arbiter is in charge of selecting a winner of all the requests, which has right to access any outport. This mechanism can be realised applying traditional round robin arbiters. If the FIFO in the next node is full, then the arbiter will not select a winner to access the requested ports. A *Mux1* unit is in charge of granting the FIFO, which holds the winner flit, to release the data flit from the register of the FIFO.

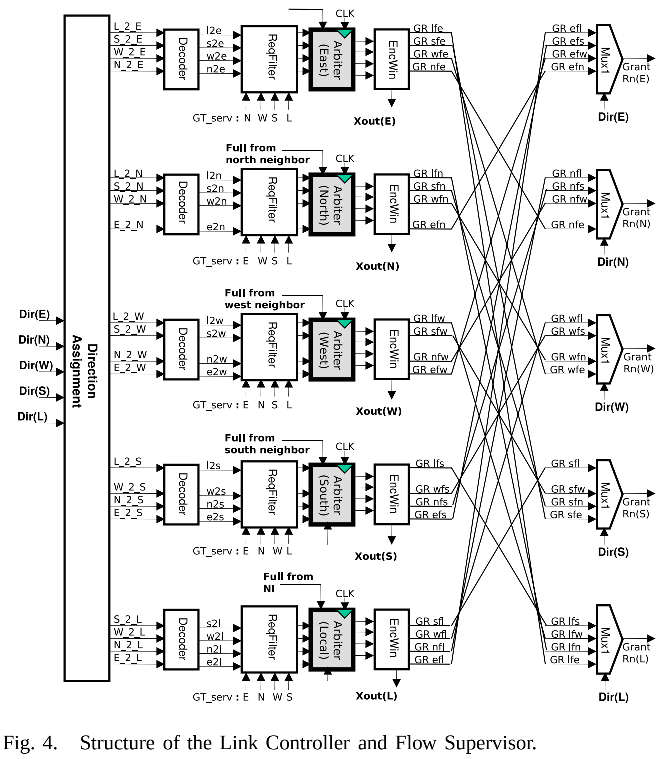


Fig. 4. Structure of the Link Controller and Flow Supervisor.

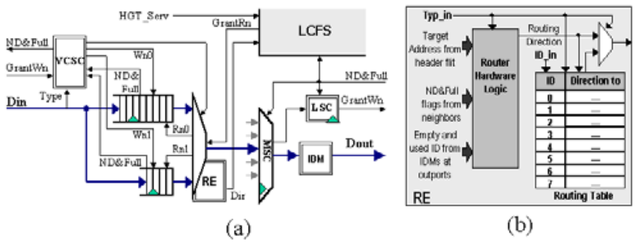


Fig. 5. (a) Virtual channels for two-level priority, and (b) routing engine.

C. Virtual Channel Control

A virtual channel state control (VCSC) unit as shown in Fig. 3(b) and Fig. 5 is used to control the flow of packets and to determine, in which FIFO the packet will be buffered by detecting the flit type field of a packet header. The HP packet will be buffered in HP FIFO. When a HP request is detected, then the VCSC unit will control a multiplexer to select an output from the HP FIFO. After the last flit of the HP packet has been forwarded and the HP FIFO is empty, then the VCSC unit will select again the output from LP FIFO.

The routing engines, which are distributed on each input port, comprises a combination of router hardware logic and an ID-based look-up table. The router hardware logic is a flexible module and contains a certain routing algorithm. It will determine the direction of packet based on information in its packet header, neighbor congestion states from downstream FIFOs and ID-slot states from IDMs in output ports. Then the routing direction is stored in the routing table of LUT module in accordance with its ID-tag address.

D. Synchronous Wormhole Packet Switching

Our NoC serves packets using a parallel pipeline wormhole packet switching technique which is operating synchronously. Fig. 6 represents our proposed two-stage pipeline switching mechanism, where a few flits flows from the west in-port in node (1,0) to an east outport in node (2,0). Transferring the flits from the FIFO buffer to the out-port, or from the out-port to next FIFO buffer requires two cycles. The first cycle is request stage. After this cycle, the RE sends direction request signals to the LCFS. The second cycle is the grant stage. After this cycle, the arbiter in the LCFS has selected the winner to access the outport by sending a *grant Rn* signal to the FIFO and a *Xout* signal to the MSC module. In the next cycle, the flit will appear in the output of the MSC module. The MSC is a multiplexor (located in crossbar switch) with a state machine mechanism. The switching is run in parallel, and contention to access the same outport is controlled by a fair flit-by-flit arbitration and a link sharing method over incoming flits. The LSC is a state machine, which controls the flow of a flit from the outport into the next FIFO buffer. If the next FIFO is full, then the LSC will not let the flit enter the next FIFO, until one space in the register of the next FIFO is free.

E. Parallel Simultaneous Crossbar Interconnect

If a single routing engine is utilized for each mesh router node, then an input selection is required to select one packet, when two or more than two packets enter the node. The routing engine will serve one by one all incoming dataflits coming into the input ports at the same time. In this case, packet latency will increase, because the other flits must wait until the single routing engine has finished to route a selected incoming flit into its required direction.

The idea of using parallel router to support parallel simultaneous crossbar interconnect is not a new topic. The works in [23], [35], [5], [13], [15] have early introduced the approach. Our NoC uses

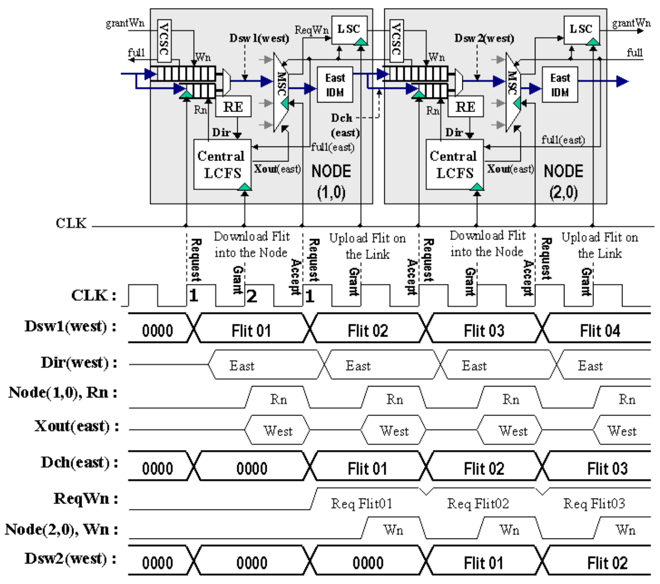


Fig. 6. Timing diagram of the synchronous wormhole switching.

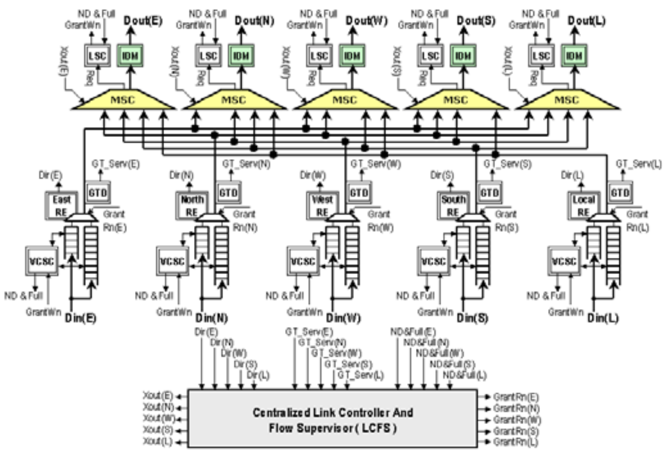


Fig. 7. The crossbar-switch and router architecture with five I/O ports.

a combination of router hardware logic and look-up table. This approach will support in-order packet delivery. Otherwise, flits of different packets can be interleaved in the same FIFO as described in Section IV-F. The Routing Engine (RE) modules are located in every input port of the mesh router node. This structure supports a ”parallel pipeline simultaneous crossbar switching”. All dataflits coming from all input ports can be routed in parallel by the routing engine (RE) at each input port and forwarded them at the same time into their required output ports. Fig. 7 shows the detail of the crossbar switch architecture (input-buffering architecture). Fig. 8 shows timing diagrams of serving incoming flits in a parallel pipeline synchronous technique. Fig. 8(a) and (b) present that all incoming flits are forwarded or served in parallel into requested output ports. In Fig. 8(c), our routing engine is used to serve the flits of the packets coming from north, west, south and local input ports with fairly round-robin arbitration. The detail of the proposed synchronous parallel pipeline switching is described in Section IV-D.

F. Locally Managed Variable Packet Identity

The routing engine (RE) uses a combination of router hardware logic and look-up tables (LUTs) allocated at each port to support

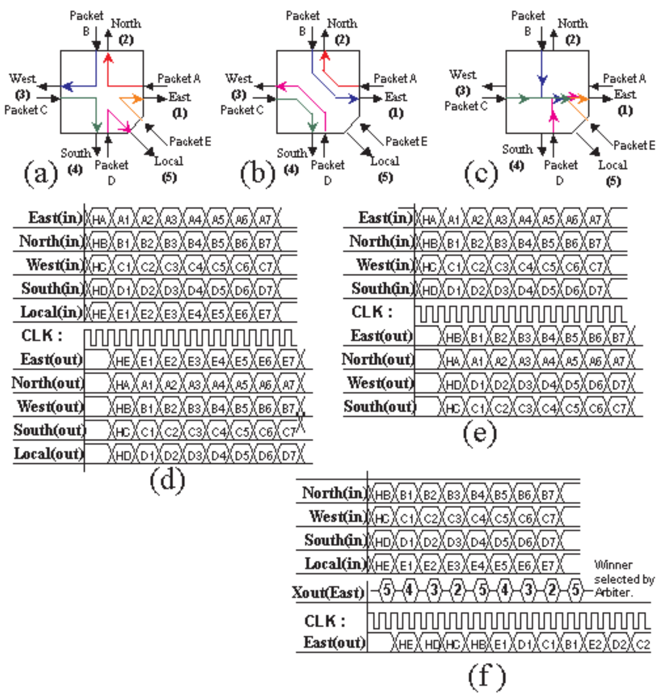


Fig. 8. Synchronous parallel simultaneous crossbar switch interconnects.

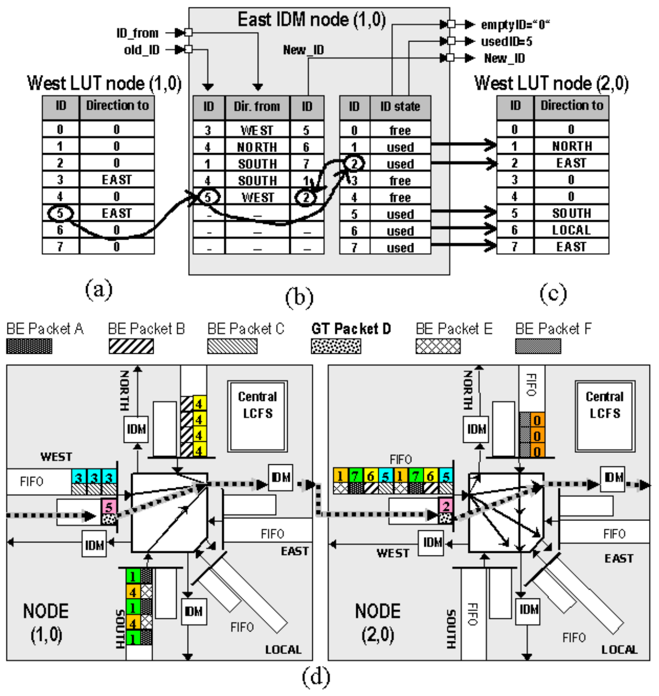


Fig. 9. The routing tables of (a) the West LUT at node (1,0), (b) East IDM at node (1,0), and (c) the West LUT at node (2,0). (d) ID-based wormhole packet flow.

parallel pipeline routing. Packet flows are controlled based on ID-tags. All flits of a packet have the same ID-tag on a certain communication link. Fig. 9(d) presents two adjacent mesh nodes with address (1,0) and (2,0). In node (1,0) there five different packets, i.e. Packet C and D with ID 3 and 5 respectively in west FIFO, Packet A and E with ID 1 and 4 in south FIFO, and Packet B with ID 4 in north FIFO.

All packets request the same outport, i.e. EAST port. Packet A, B, C, E and F are BE packets, while packet D (with ID 5 in node(1,0) and ID 2 in node (2,0) is HP packet. Packet D has higher priority than the other LP packets to access the link. The other LP packets must wait for a while until Packet D has closed the reservation, i.e. the last flit of Packet D has deleted the HP state signal in VCSC as explained in Section IV-C. Therefore the HP packet is suitable or more effective for small until medium size hard real-time messages.

The router hardware logic computes the required routing direction based on the target address information in the packet header and the underlying routing algorithm. The routing direction of the packet is then copied into the routing table registers of the LUT in accordance with its ID-tag and direction. An example is shown in Fig. 9(a).

Assuming that the headers of Packet C and D in the west FIFO have been evaluated and their computed directions have been copied into the routing table registers of the LUT. Then each flit belongs to Packet C with ID 3 and Packet D with ID 5 will be then routed to EAST and SOUTH respectively. In other words, a payload flit, which has the same ID-tag as the previous header, will be routed (switched) based on its ID-tag.

The IDM unit will update new ID for new packet flowing through the outport. The IDM provides implicitly space-slot for packets, and will guarantee, that different packets will have a different ID-tag in a certain FIFO segment over the NoC. For a 2-D 4x4 mesh-based NoC, the IDM provides 8 ID-tags (8 virtual space slots) for each link. Certainly, for larger size NoC, number of available ID-tag can be extended.

The IDM will manage the ID allocation, before a new different packet enters the next FIFO buffer. Fig. 9(b) illustrates the functionality of IDM in accordance with packet flows shown in Fig. 9(d). The packets are classified based on their ID and from which input port they come from. For a new packet header (Packet C from west input port with ID 3 for example), the IDM will search for a 'free' ID. If the free ID has been found (i.e. ID 5 for example), then the old ID of the packet header (ID 3) is replaced by the new ID (ID 5), and the state of the ID is set to 'used'. There is also a possibility that packets coming from different input ports have the same ID-tag, e.g. Packet E from south and B from north with ID 4. The IDM will manage the ID in such as way that they will have different IDs in the next FIFO (e.g. new ID 6 for Packet B and new ID 1 for Packet E. The IDM will then save the information in the register tables. For payload flits following the header flit of the packets, their IDs will be replaced automatically by using look-table mechanism.

If there is no more available ID in the IDM, then new packet cannot be forwarded to the outport. After the last flit of the packet flows through the LUT and the IDM, all informations related to its ID-tag will be deleted from the tables. Our ID-tag based adaptive routing mechanism will also guarantee in-order-delivery of a packet.

G. Link-Level Flow and Automatic Injection-Rate Control

Because the links are shared, bottleneck phenomena can potentially occur in buffers of router channels. This situation is also called 'hotspot', or congestion state, i.e. the situation where buffers tend to become full. Our NoC is facilitated with a control mechanism for link-level flit flow control to avoid packets entering full buffers, and injection rate control mechanism, which is implemented to follow automatically the accepted injection-rate based on traffic conditions of the network. Fig. 10 shows how this mechanism works. Packet A, B and C are injected from nodes (0,1), (0,0) and (1,0), then ejected from nodes (2,0), (1,1) and (2,1) respectively. Assuming that all packets are LP packets, therefore they are buffered in the LP FIFOs. Packet B

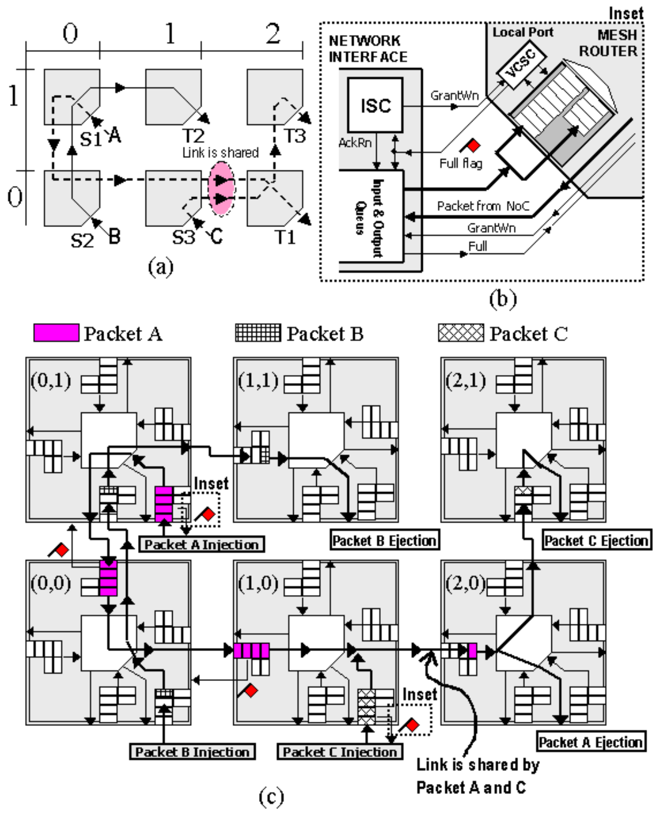


Fig. 10. Packet flows and automatic injection rate control.

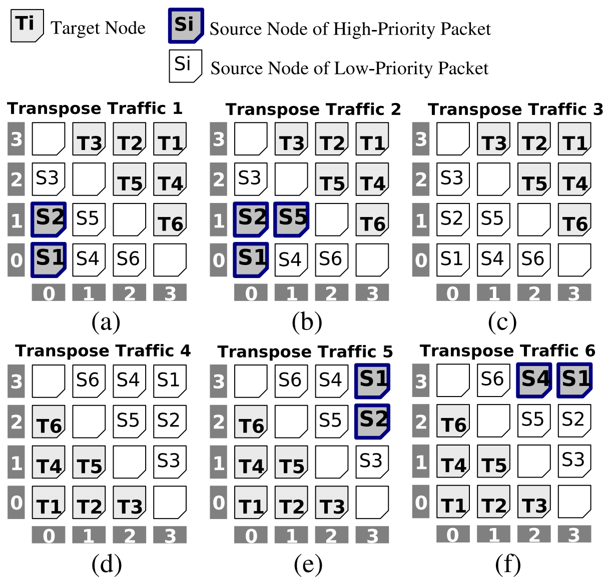


Fig. 11. Selected traffic scenario.

enjoys 100% of the link bandwidth. Hence it is always injected from node (0,0) with maximum injection-rate. Meanwhile, packet A and C share the link between node (1,0) and (2,0). After a few cycle, west and local FIFO at node (1,0) are congested (full). As a consequence, the local FIFO at node (0,0) will be also full, because the arbiter for east outport at node (1,0) must select fairly the winner to access the port. The situation is not a problem in our NoC, because LSC as described in Section IV-D will control the overflow. It is important to note, that although the FIFO is full, it is still possible for another packet to insert its flit, as long as there is still available free ID-tag, and after a few cycles, there is again one free space in the FIFO.

ISC (Injection State Control) unit in the network interface (NI) (Fig. 10(c)) controls automatically the injection rate. If the local FIFO is full then the ISC will not grant FIFO to accept the flit and will not also permit the Input Queue in the NI to release the flit, until there is free space in the local FIFO. This mechanism will automatically reduce the injection rate.

V. Experimental Results

The performance of the proposed NoC prototype using west-first routing algorithm is evaluated by using two different traffic scenarios as shown in Fig. 11. Matrix-transpose traffic scenarios as depicted in Fig. 11 are selected. Fig. 11(a) shows a traffic scenario, where 6 different messages are injected from lower-left region and accepted from upper-right region. While in Fig. 11(b), 6 different messages are injected from upper-right region into lower-left region. Source and Target nodes are identified by letters S and T respectively followed by numerical identifiers. Source nodes with in grey color are the nodes, where HP packets are injected. The objectives of both transpose traffic scenarios are to observe how the NoC gives different services for low and high priority messages and to control their flow in the network.

In these experiments, 6 medium size messages are used to evaluate the performance of the NoC. Each injected message consists of 128 flits (1 header flit followed by 127 payload flits). Therefore, in the traffic scenarios, total number of 768 flits (6 x 128) are injected in the on-chip network. Each flit in the packet is numbered in-order, thus it is easy for us to check packet-loss, packet integrity and out-of-order delivery. In general, the number of required cycles (in cycle periods) to transmit the last flit of each independent packet is measured. This measurement is started when headers of the packets are injected to source node. The required cycle periods to transmit the last flit of each packet are shown in Table I until Table VIII for each traffic scenario. The measurement shows the number of cycles required to transmit the last flit of each message from the source node to the target node.

In these experiments, 6 medium size messsages are used to evaluate the performance of our NoC. Each injected packet consists of 128 flits (1 header flit followed by 127 payload flits). Therefore, in the traffic scenarios, total number of 768 flits (6 x 128) are injected in the NoC. Each flit in the packet is numbered in-order, thus it is easy for us to check packet-loss, packet integrity and out-of-order delivery. In general, number of required cycle (in cycle period) to transmit the last flit of each independent packet is measured. This measurement is started when headers of the packets are injected to source node. The required cycle periods to transmit the last flit of each packet are presented from Table I until Table VIII for each traffic scenario. The measurement includes the number of hop cycles from the source node until the target node.

In transpose traffic scenarios 1, 3, 4, 5 and 6, all packets are injected at the same time, and required cycles to transmit the last flit of each packet are measured in cycle period. While in traffic scenario 2, the injections of a few packet are delayed. In traffic scenario 2, with experiment 1, all packets are injected at the same time (see Table II and Fig. 12(a)). In traffic scenario 2, with experiment 2, the injection start time of HP packet P5 is delayed 50 cycle periods. (see Table III and Fig. 12(b)). In traffic scenario 2, with experiment 3, the injection start times of HP packet P2 and LP packet P4 are delayed 50 and 60 cycle periods respectively. (see Table IV and Fig. 12(c)).

Paths from source to target nodes setup by each packet are shown in Fig. 12 for traffic scenarios 1 and 2, and in Fig. 13 for traffic

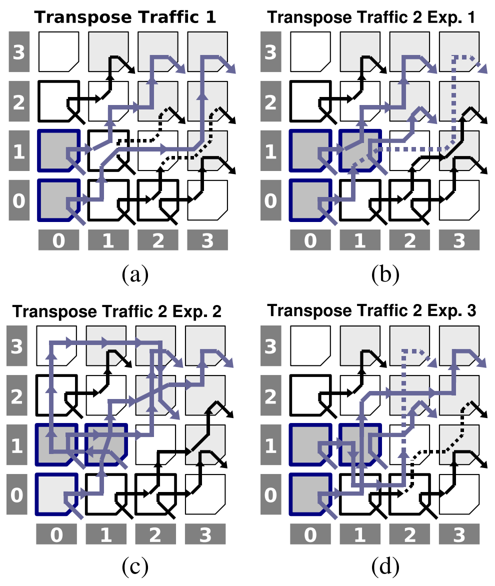

Transpose Traffic 1
(a)
Transpose Traffic 2 Exp. 1
(b)
Transpose Traffic 2 Exp. 2
(c)
Transpose Traffic 2 Exp. 3
(d)

Fig. 12. Setup paths in traffic (a) scenario 1, (b) scenario 2 experiment 1, (c) scenario 2 experiment 2, and (d) scenario 2 experiment 3.

scenarios 3, 4, 5 and 6. The gray arrow lines represent paths setup by HP packets, while the black arrow lines represent paths setup by LP packets. The dashed lines in all figures present that the packet must wait for a moment to let HP packet flowing through output link until its last flit has been forwarded. HP packet will select non minimal direction (e.g. P5 injected at node (1,1) shown in Fig. 12(c)), if all possible direction have been reserved by the other HP packets. Otherwise, a HP packet must also wait for a while, if non-minimal routing is not possible (because of prohibited turn models) and the requested link is used by another HP packet as shown in Fig. 12(d).

Examples of timing diagram of the simulation results are presented in Fig. 14 for traffic scenario 4, and in Fig. 15 for traffic scenario 5. In the traffic scenario 4 and 5, all packets are transposed from upper-right into lower-left region. In this case (using the adaptive west-first routing algorithm), all packet will be routed firstly into west direction and then into south direction.

In traffic scenario 4, all packets are LP packets. Hence, the last flit of P1 injected at node (3,3) will be the last flit, which arrives the target node, because the distance of its target node is the farthest compared to the other packets. Although the source-target node distances of P2 and P4 are the same, required cycle period to transmit P4 is more than P2, because P4 shares a few links with P1 and P6, while P2 shares a few links with only Packet P5. Because some links are shared among packets (except Packet P3), the injection rates of P1, P2, P4, P5 and P6 are reduced automatically (see Fig. 14). However, there is no packet-loss in this situation, because injection rates are automatically controlled by ISC module in network interface, when network segments are saturated as early described in Section IV-G.

In traffic scenario 5, P1 and P2 injected at node (3,3) and (3,2) are HP packets. The P1 and P2 are always injected from source node with the maximum injection rate (see Fig. 15). Therefore, latency bounds of P1 and P2 are lower than latency bounds shown in traffic scenario 4, where P1 and P2 are LP packets.

VI. Conclusions and Future Works

A. Concluding Remarks

The NoC prototypes using an adaptive west-first routing algorithm with only LP service and combined LP and HP services have been

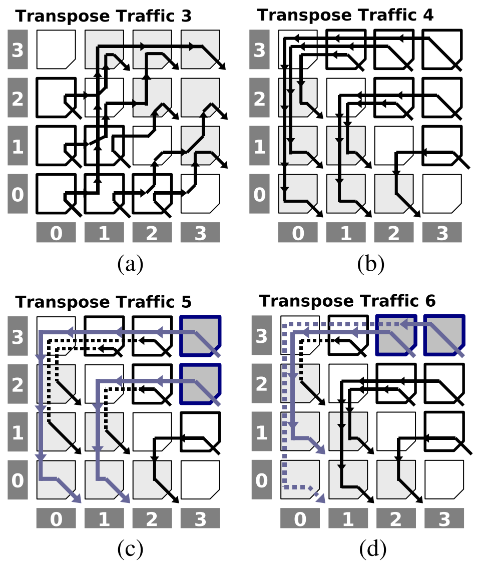


Fig. 13. Setup paths in traffic (a) scenario 3, (b) scenario 4, (c) scenario 5, and (d) scenario 6.

TABLE I
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 1

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	HP	LP	LP	LP	LP
Accept Last Flit (Cycle)	282	274	266	532	524	266

TABLE II
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 2 EXPERIMENT 1

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	HP	LP	LP	HP	LP
Injection start at Cycle	0	0	0	0	0	0
Accept Last Flit (Cycle)	530	274	266	274	266	266

TABLE III
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 2 EXPERIMENT 2

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	HP	LP	LP	HP	LP
Injection start at Cycle	0	0	0	0	50	0
Accept Last Flit (Cycle)	282	274	266	276	332	266

TABLE IV
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 2 EXPERIMENT 3

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	HP	LP	LP	HP	LP
Injection start at Cycle	0	50	0	60	0	0
Accept Last Flit (Cycle)	282	324	266	526	266	266

TABLE V
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 3

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	LP	LP	LP	LP	LP	LP
Accept Last Flit (Cycle)	534	526	514	274	266	266

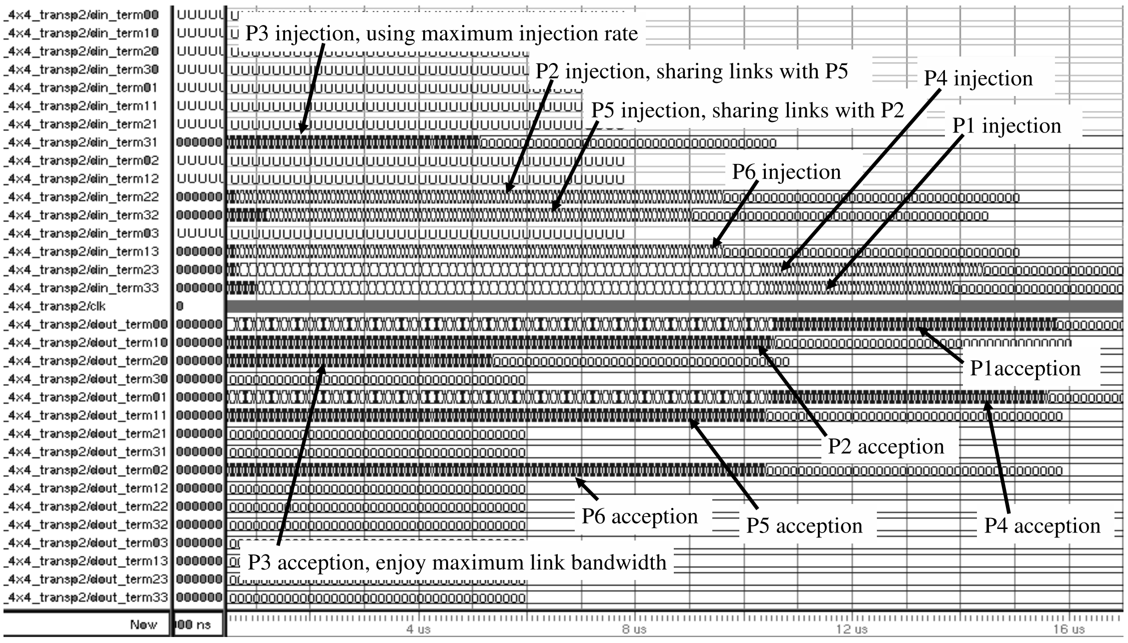


Fig. 14. Timing diagram of the simulation for traffic scenario 4.

TABLE VI
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 4

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	LP	LP	LP	LP	LP	LP
Accept Last Flit (Cycle)	786	526	266	778	518	518

TABLE VII
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 5

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	HP	LP	LP	LP	LP
Accept Last Flit (Cycle)	282	274	266	784	524	774

synthesized using UMC 180nm standard-cell technology. Table IX summarizes the total cells area and targeted allowing working frequency of both prototypes. The depth of the LP FIFO and the HP FIFO is set to 4 in this case. It looks that the area overhead to implement the NoC combining LP and HP message services is about 48 % over the NoC with only LP message service. While the working frequency is reduced from 400 MHz to 330 MHz (18 % reduction). Fig. 16 shows circuit layout of the NoC mesh router prototype with two-level priority service in a 2x2 topology. Four areas of the mesh router node (nodes (0,0), (0,1), (1,0) and (1,1)) are shown in the figure.

As long as there is no contention between two or more HP packets to access an output port in a mesh router, and HP packets are not routed into non minimal direction (away from target node), the

TABLE VIII
MEASUREMENT OF REQUIRED CYCLES (IN CYCLE PERIOD) TO TRANSMIT LAST FLITS OF EACH PACKET FOR TRAFFIC SCENARIO 6

Packet Number	P1	P2	P3	P4	P5	P6
Packet Type	HP	LP	LP	HP	LP	LP
Accept Last Flit (Cycle)	534	526	266	274	518	780

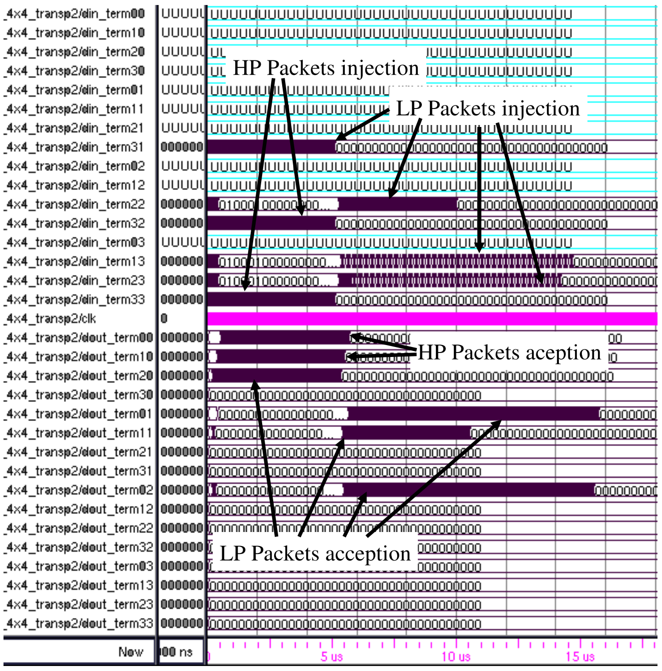


Fig. 15. Timing diagram of the simulation for traffic scenario 5.

latency of the completion bound of the HP packets is predictable (the time on when the last flit of the packet arrives the target node is predictable). If HP packet is routed with non minimal routing direction, then the latency of the completion bound is higher than its predicted latency. However the HP packet can be still injected with maximum rate, as far as the HP packet can still find optimal paths from source to target nodes, which are free from reservation by the other HP packets.

Our NoC prototype combining LP and HP best effort services can guarantee lossless packet completion and in-order message delivery. Out-of-order problem can be avoided because of the used packet

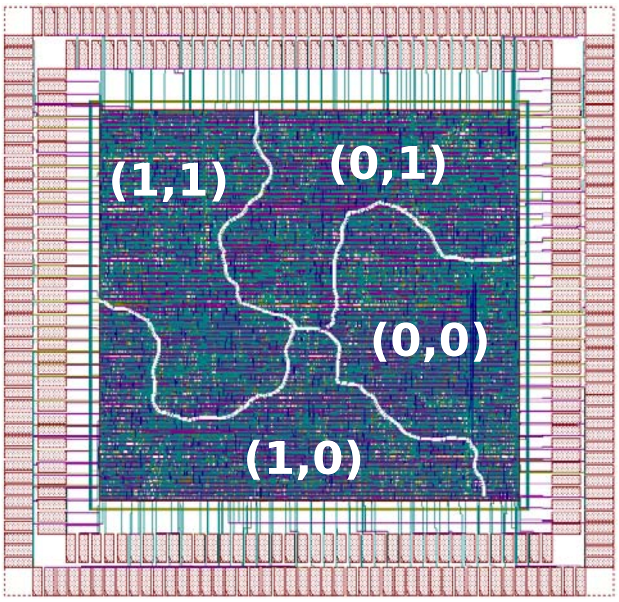


Fig. 16. Circuit layout of the NoC in 2x2 mesh topology using 180-nm standard-cell technology.

TABLE IX
LOGIC SYNTHESIS RESULTS.

Service Provided	LP Service	LP+HP Service	Overhead
Total cells area	0.218 mm^2	0.323 mm^2	48 %
Target frequency	400 MHz	330 MHz	-18 %

format and the routing organization between the router hardware logic and the routing table in the routing engine at each input port. In the current implementation, the messages injected as HP packet will consume absolutely 100% bandwidth of their reserved links if they are injected with maximum injection rate. Therefore, the HP packets having critial time in any application should be small or medium-size messages. Else their injection rate must be controlled or reduced in network interface, so that the HP queue will be empty in any instant time. Hence, at that instant time, LP packet, which are buffered in the LP queue at the same input port, can be forwarded to their requested output port.

Table X present a comparison between our NoC and other two NoCs that use virtual channels to provide additional services. The BEQ and GSQ stand for best-effort queue and guaranteed-service queue. While LPQ and HPQ stand for low-priority queue and high-priority queue. The area overhead in our NoC is due to the use of larger size technology (180-nm), and the use of ID-management unit at each output port on the router. However, the area cost is paid for the more flexible wormhole switching method that has been presented in Section IV-F.

TABLE X
NOCS COMPARISON.

NoC Prototype	Æthereal [23]	DSPIN [13]	Our NoC
Techn. size	130 nm	90 nm	180 nm
Total cells area	0.260 mm^2	0.082 mm^2	0.323 mm^2
Data frequency	500 MHz	500 MHz	330 MHz
FIFO/Queue Depth	BEQ is 8 GSQ is 1	BEQ is 8 GSQ is 4	LPQ is 4 HPQ is 4

B. Future Work

The HP message service is more suitable for critical time-constraint data, where the sizes of the messages are small or medium. Because if HP messages are injected with maximum injection rate and reserve a link for a very long time period, then LP or other HP packets, which also require the reserved links, will be also stagnant for a very long time until the HP packets close the link reservation. If the traffic of a certain real-time application is predictable, then this problem can be handled by using an optimal resource placement technique during the application mapping in system level.

Our current NoC prototype with two-level priority data delivery service uses only a best-effort data delivery method. For future investigation, a new NoC variant with combined Best-Effort (connectionless), and Guaranteed-Throughput (connection-oriented) will be developed. In the connection-oriented service, packet headers will be injected firstly to find the possible links to target node. After the header arrives the target node, a response flit or feedback information flit will be sent back to the source node. The response flit will bring an information wether the connection between the source and the target node is successfully established. If the connection has been setup, then the source node starts injecting the message or stream into the NoC. At the end of the data injection process, a tail flit is then sent to the network to close the connection.

ACKNOWLEDGEMENT

The authors would like to thank DAAD (*Deutscher Akademischer Austausch-Dienst*, German Academic Exchange Service) for supporting the first author to pursue PhD-degree at TU Darmstadt Germany.

REFERENCES

[1] P. Martin, “Design of a Virtual Component Neutral Network-on-Chip Transaction Layer,” *Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE’05)*, pp. 336-337, 2005.
[2] D. Wingard, “MicroNetwork-Based Integration for SOCs,” *Proc. Design Automation Conf. (DAC’01)*, pp. 673-677, 2001.
[3] L. Benini and G. De Micheli, “Networks on Chips: A New SoC Paradigm,” *IEEE Computer*, vol. 35, pp. 70-78, Jan. 2002.
[4] A. Jantsch and H. Tenhunen, *Networks on Chip*, Kluwer Academic Publisher, Hingham, MA, USA, 2003.
[5] M. Millberg, E. Nilsson, R. Thid and A. Jantsch, “Guaranteed Bandwidth using Looped Containers in Temporally Disjoint Networks within the Nostrum Network on Chip,” *Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE’04)*, pp. 890-895, 2004.
[6] D. Wiklund and D. Liu, “SoCBUS: Switched Network on Chip for Hard Real TIme Embedded Systems,” *Proc. IEEE Int’l Parallel and Distributed Processing Symposium (IPDPS’03)*, 8 pp., 2003.
[7] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, et. al., “The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs,” *IEEE Micro*, vol. 22, issue 2, pp. 25-35, Mar-Apr. 2002.
[8] C. Hilton and B. Nelson, “PNOC: a flexible circuit-switched NoC for FPGA-based systems,” *IEE Proc. Computers and Digital Techniques*, vol. 153, no.3, pp. 181-188, May 2006.
[9] S. Kumar, A. Jantsch, J. -K. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrja and A. Hemani, “A Network on Chip Architecture and Design Methodology,” *Proc. IEEE Computer Society Annual Symposium on VLSI*, pp. 105-112, 2002.
[10] M. K. -F. Schäfer, T. Hollstein, H. Zimmer, M. Glesner, “Deadlock-Free Routing and Component Placement for Irregular Mesh-based Network-on-Chip,” *IEEE/ACM Int’l Conf. on CAD (ICCAD’05)*, pp. 238-245, 2005.
[11] F. Karim, A. Nguyen and S. Dey, “An Interconnect Architecture for Networking Systems on Chips,” *IEEE Micro*, vol. 22, issue 5, pp. 36-45, Sept-Oct. 2002.

[12] P. Guerrier and A. Greiner, "A Generic Architecture for On-Chip Packet-Switched Interconnection," *Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE'00),* pp. 250-256, 2000.

[13] I. M. Panades, A. Greiner and A. Sheibanyrad, "A Low Cost Network-on-Chip with Guaranteed Service Well Suited to the GALS Approach," *Proc. the 1st Int'l Conf. and Workshop on Nano-Networks),* pp. 1-5, 2006.

[14] T. A. Bartic, J. -Y. Mignolet, V. Nollet, T. Marescaux, D. Verkest, S. Vernalde and R. Lauwereins, "Topology adaptive network-on-chip design and implementation," *IEE Proc. Computers and Digital Techniques,* vol. 152, no.4, pp. 467-472, July 2005.

[15] L. Benini and D. Bertozzi, "Network-on-chip architectures and design methods," *IEE Proc. Computers and Digital Techniques,* vol. 152, no.2, pp. 261-272, Mar. 2005.

[16] J. Xu, W. Wolf, J. Henkel and S. Chakradhar, "A Design Methodology for application-Specific Networks-on-Chip," *ACM Trans. on Embedded Computing Systems,* vol. 5, no. 2, pp. 263-280, May 2006.

[17] J. Bainbridge and S. Furber, "Chain: A Delay-Insensitive Chip Area Interconnect," *IEEE Micro,* vol. 22, issue 5, pp. 16-23, Sept-Oct. 2002.

[18] M. Amde, T. Felicijan, A. Efthymiou, D. Edwards and L. Lavagno, "Asynchronous on-chip networks," *IEE Proc. Computers and Digital Techniques,* vol. 152, no. 2, pp. 273-283, Mar. 2005.

[19] M. Sgroi, M. Sheets, K. Keutzer, S. Malik, J. Rabaey and A. S. Vincentelli, "Addressing the System-on-a-Chip Interconnect Woes Through Communication-Based Design," *The 38th ACM Design Automation Conf. (DAC'01),* pp. 667-672, 2001.

[20] I. Saastamoinen, D. S. -Tortosa and J. Nurmi, "Interconnect IP Node for Future System-on-Chip Designs," *The 1st IEEE Int'l Workshop on Electronic Design, Test and Applications (DELTA'02),* pp. 116-120, 2002.

[21] E. Beigné, F. Clermidy, P. Vivet, A. Clouard and M. Renaudin, "An Asynchronous NOC Architecture Providing Low Latency Service and its Muti-level Design Framework," *Proc. the 11th IEEE Int'l Symp. on Asynchronous Circuits and Systems,* pp. 54-63, 2005.

[22] T. Bjerregaard and J. Sparsø, "Implementation of guaranteed services in the MANGO clockless network-on-chip," *IEE Proc. Computers and Digital Techniques,* vol. 153, no.4, pp. 217-229, July 2006.

[23] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage and E. Waterlander, "Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip," *IEE Proc. Computers and Digital Techniques,* vol. 150, no. 5, pp. 294-302, Sep. 2003.

[24] S. M. Parker et. al., "SpaceWire: Links, Nodes, Routers and Networks," *European Cooperation for Space Standardization, Standard No. ECSS-E50-12A,Issue 1,* January 2003.

[25] Intel Corp., "Overview, Installing and Using Priority Packet II," available online: http://www.intel.com/support/network/adapter/ppack/sb/cs-013750.htm.

[26] Free Patents online, "Transmission of high-priority, real-time traffic on low-speed communications links", available online: http://www.freepatentsonline.com/EP1128612.html.

[27] Patent Storm, US Patent 7120113, "Systems and methods for limiting low priority traffic from blocking high priority traffic," available online: http://www.patentstorm.us/patents/7120113.html.

[28] J. S. Choi and C. K. Un, "Delay performance of an input queueing packet switch with twopriority classes," *IEE Proceeding, Communications,* Volume 145, Issue 3, pp. 141–144, Jun 1998.

[29] J. Duato, S. Yalamanchili and L. Ni, *Interconnection Networks: An Engineering Approach*, Revised Printing, San Fransisco, USA: Morgan Kaufmann Publishers, 2003.

[30] C. J. Glass and L. M. Ni "The Turn Model for Adaptive Routing, " *The 19th Int'l Symposium on Computer Architecture,* pp. 278-287, 1992.

[31] C. J. Glass and L. M. Ni "Adaptive Routing in Mesh-Connected Networks, " *The 12th Int'l Conference on Distributed Computing Systems,* pp. 12-19, 1992.

[32] J. Kim, D. Park, N. Vijaykrishnan and C. R. Das, "A Low Latency Router Supporting Adaptivity for On-Chip Interconnects, " *ACM Design Automation Conf. (DAC'05),* pp. 559-564, 2005.

[33] D. Seo, A. Ali, W. -T. Lim and N. Rafique, "Near-Optimal Worst-case Throughput Routing for Two-Dimensional Mesh Networks, " *The 32nd Int'l Symp. on Computer Architecture,* pp. 432-443, 2005.

[34] G. -M. Chiu, "The Odd-Even Turn Model for Adaptive Routing, " *IEEE Trans. on Parallel and Distributed Systems,* vol. 11, no. 7, pp. 729-738, July 2000.

[35] J. Hu and R. Marculescu, "DyAD - Smart Routing for Network-on-Chip, " *ACM Design Automation Conf.,* pp. 260-263, 2004.

[36] G. Ascia, V. Catania, M. Palesi and D. Patti "Neighbor-on-Path: A New Selection Strategy for On-Chip Networks, " *IEEE/ACM/IFIP Workshop on Embedded Systems for Real-Time Multimedia,* pp. 79-84, 2006.