Network-on-Chip with Guaranteed-Bandwidth Data Communication Service
Interconnect Platform for Computer Vision and Multimedia Applications on Many Core Processors

Faizal Arya Samman
Department of Electrical Engineering, University of Hasanuddin at Makassar
Kampus Gowa, Jl. Poros Malino Km. 20, Borongloe 92172, Bontomaranu, Gowa
Email: faizalas@unhas.ac.id

Abstract—A network-on-chip (NoC), having guaranteed-throughput (GT) or guaranteed bandwidth service by using a flexible method to establish a connection-oriented data communication at runtime, is presented in this paper. The GT packets can share communication link with a flexible manner, where flits belonging to the same packet will have the same local identity-tag (ID-tag). The ID-tags of each packet will vary locally along communication links, and are organized with an ID-tag mapping management unit, which is implemented at each output port of the on-chip routers. There is no need for a specific algorithm for finding a conflict-free scheduling as commonly used in the TDM-based methods that use time-slot allocation technique. The contention problem is solved with the hardware solution based on the locally organized message identity (ID). This guaranteed bandwidth/throughput service will provide a good interconnect platform for many core processor systems running computer vision and multimedia applications with better performance.

Keywords—Guaranteed Bandwidth, Network-on-Chip, Multimedia, Computer Vision, Many Core Processors

I. INTRODUCTION

Quality of Service has been an important issue in an internetworking data communication to provide a better service for certain data traffics. Some network-on-chip (NoC) prototypes have also considered this issue in the NoC context to provide a specific service for traffics requiring a guaranteed throughput and latency such as video stream data. The effort to provide the better service will always face a problem on how to manage contentions (conflicts) between different types of packets such that they can share each communication link in the NoC, while maintaining application performance. The quality of service can be made through separate virtual channels or even completely partitioned subnetworks [1].

In internetworking communities, some communication protocols have been introduced to guarantee the quality of service. In general, such protocols can be characterized as connectionless and connection-oriented protocols. In general, connection-oriented services provide some levels of delivery guarantees, whereas connectionless services do not [2]. We can interpret also a packet being injected to the network with the connectionless protocol as a best-effort (BE) packet, and a packet being injected with the connection-oriented protocol as a guaranteed-throughput (GT) packet.

In order to provide a specific quality of service for certain traffics, two approaches are proposed i.e., resource reservation and priority-based scheduling strategies. The resource reservation strategy can provide a hard performance guarantee, but communication resource utilization may be lower. While, priority-based strategy can achieve better communication resource utilization, but the performance guarantee is soft [3]. Regarding the connectionless and connection-oriented services mentioned before, the resource reservation strategy will be implemented using the connection-oriented service, while the priority-based scheduling method can be made using the connectionless service.

Data communication between IP cores in the NoC-based multi-processor systems can be realized in general using circuit switching, packet switching and wormhole switching method. A virtual cut-through (VCT) switching method is special case of the packet switching where a packet can cut-through i.e., the packet can be switched out soon after a routing decision has been made without waiting for the packet to fully store in a buffer of the NoC router. Some recently published chip multiprocessor (CMP) systems with interconnected processing elements such Cell EIB [4], Tile64 [5], TRIPS [6], Teraflops [7] and SCC NoC [8] uses the packet switching method. The circuit-switching based on the Time-Division Multiplexing (TDM) method has been used by some NoC proposals. Examples of them are [3], and ÅEtheral [10].

The use of switching methodology for a NoC-based system can determine how the guaranteed-service can be feasible implemented in the NoC. The circuit switching method for instance enables the feasible implementation of a TDM-based guaranteed-service through the communication resource reservation approach. This paper will present an extensive use of the wormhole switching method, in which the wormhole packets can be interleaved (cut-through) at flit-level with different packets on the same link [16]. Instead of using time slots, we use local ID slots to solve the BE-GT packet conflicts.

The communication resources reservation approach can be used because of the support of the local message ID organization technique. Our NoC uses both the connectionless (BE) and connection-oriented with GT protocols. The difference between both protocols is not much and explained in the following. The BE data traffics are injected to the NoC soon to follow the BE header, while the BE header flit is making communication resource reservation. The source node does not wait for a response from the destination node to know whether the connection has been successfully established or not. The GT data traffics or streams are injected after the connection establishment is successfully made by the GT header flit by reading an information in the accepted response flit sent by the destination node. The communication links are shared by all GT packets fairly, and can also be shared with BE packets by controlling the injection rate of the GT packets in such a way that there will be an instant time used by the BE packets being routed to the requested links.

The paper is organized in the following sections. Section III presents some works related to our current research and introduces a novel methodology to implement and to combine the connectionless and connection-oriented routing protocols with guaranteed-bandwidth service. Section IV presents the flexibility of our methodology to establish connection at runtime (during application execution) compared with the existing time-slot-based TDM switching method. Features and characteristics of the proposed NoC microarchitecture including the hardware solution to realize the connection-oriented guaranteed-service is exhibited in Section V. Section IX presents experimental results to observe the effectiveness of our methodology to combine BE and GT traffics in our NoC. Finally, Section X gives concluding remarks and future research directions.

978-1-5090-5548-7/16/$31.00 © 2016 IEEE
II. RELATED WORKS

The work in [11] proposes a unified Mapping, Routing and Slot Allocation (UMARS+) algorithm for NoCs supporting Best-Effort and Guaranteed Services. The proposed algorithm couples path selection, mapping of application cores and channel time-slot allocation to minimize a network required to meet the constraints of the application. The work in [3] has also presented an interesting algorithm and methodology to make time-slot-based link configuration and scheduling. The work focuses on TDM Virtual Circuit (VC) and address a multidisconnection configuration problem. The VC configuration is the background idea of a connection-oriented communication service between communicating cores in a packet switched network. For a static case, the VC configurations are computed off-line. The authors have mentioned that the methodology can also be used for dynamic and semi-static cases. However, the further work has not been presented in detail so far.

The design flow of \( \mu \)-Spidergon NoC [12] proposes also an alternative solution offering TDMA-based guaranteed-service for data traffics in the context of NoC with different clock area and skew (GALS Globally Asynchronous Locally Synchronous context). A time router and TDMA synchronizer are included in the design flow based on a concept of sub-NoC compositions. The main feature of this approach is the use the GALS-oriented which means that no global clock is required. The work in [13] has implemented the concepts of spatial division multiplexing (SDM) for guaranteed throughput NoCs. Beside the advantageous mentioned in the work, the SDM-based method has drawbacks also, i.e. the difficulty to implement end-to-end flow control, larger area overhead for BE service implementation and the need for more complex switch control.

The aforementioned TDM-based switching methods that use time slots to allocate each packet must be implemented with a conflict-free routing and scheduling. As a result, the utilization of the communication resources is not optimal. Link scheduling at runtime during application executions based on the time-slot allocation technique is difficult, and the probability that a packet fails to establish connection is necessary to store the information which can lead also to an area overhead. Furthermore, the time slot allocation algorithms could give rise to a few time-overhead. Before an application is executed, the time-slot allocation algorithm must be run at compile time or possibly at design time. Afterwards, the results of the time-slot allocation must be distributed in the time-slots of each link or router. Hence, a reconfiguration unit on each router is necessary to store the information which can lead also to an area overhead. Furthermore, the time slot allocation algorithms could give rise to a few time-overhead. Before an application is executed, the time-slot allocation algorithm must be run at compile time or possibly at design time. Afterwards, the results of the time-slot allocation must be distributed in the time-slots of each link or router. Hence, a reconfiguration unit on each router is necessary to store the information which can lead also to an area overhead. Furthermore, the time slot allocation algorithms could give rise to a few time-overhead.

In addition, the works in [11], [3] and in [12] have presented the off-line algorithms (high-effort software solutions) for finding conflict-free routing based on time-slot allocation on every communication link. In this paper, we propose only a simple and flexible hardware solution by using the local ID slots organization technique, in which the routing conflicts can be simply managed without the off-line software.

III. CONTRIBUTION

Our previous paper has presented a NoC that only provides the connection-oriented guaranteed-through service [18]. This paper proposes a new contribution to combine connection-oriented (guaranteed-throughput) and connectionless (best-effort) services based on locally organized message identity (ID).

The main idea of the two types of packet is the use of the dynamic local ID management and control [14]. Instead of using time slots and preventing packet conflicts, we use local ID slots to optimize communication resources utilization and to ease runtime connection configuration. In our approach, conflicts between flits of different best-effort and guaranteed-throughput messages to share link are allowed, but they are controlled and managed by using a local ID-slot mapping and management technique.

IV. RUNTIME CONNECTION SETUP METHODS

\( \Phi \)etheral NoC [10] utilizes a Slot Table to avoid packet contents on a link. It divides up bandwidth per link between connections, and switch data to a correct output. Every slot table \( T \) consists of \( S \) time slots and \( N \) number of router output ports. Synchronicity is made based on incremented time slot, where all routers in the network are in the same fixed-duration slot. In a slot \( s \) at most one block of data can be taken per input port or forwarded per output port. In the next slot, \( s+1 \)%, the taken packet block are written to their appropriate output ports. Hence, the packet blocks propagate in a store-and-forward fashion.

Nevertheless, this time slot method has disadvantage. For instance, (TDMA) scheduling method has higher probability of failure to set up connection at runtime, even though there is still free time slots in the considered outgoing port. This failure happens for example when a contention occurs. In order to overcome the contention problem, NoCs which use time-slot-based TDM scheduling to provide guaranteed-service should provide a time-slot allocation algorithm to achieve a contention-free routing. Such algorithms for instances have been introduced in [3], [11] and in [12].

Instead of including a time slot allocation algorithm in design flow that must be run during compile time (before application execution time) and overloads the complexity of functional task application mapping algorithm, a more flexible approach by introducing a runtime ID-slot-based scheduling method that uses ID-Tag Mapping Management (IDM) units on every link is proposed in this paper. It provides guaranteed-service and can optimize dynamically the link bandwidth utilization of the NoC.

V. XHInoC CHARACTERISTICS

A. Network Topology

Fig. 1 presents an example a chip multiprocessor (CMP) system on a 2D 4x4 mesh planar NoC topology. Physically, the mesh planar NoC is divided into the X+ and the X− subnetworks. Compared with standard mesh router, the NoC has two pairs of vertical links connecting South and North sides of the router, i.e. South1 and North1 links on the left side used to route packets through the X+ subnetwork and South2 and North2 links on the right side used to routes packet through the X− subnetwork. The additional links increase bandwidth capacity of the router and allow us to implement a 2D planar adaptive routing algorithm. In a mesh standard, for \( M \times N \) network size, there are \( N \times (M - 1) + M \times (N - 1) \) available full-duplex links, where \( M \) is the width and \( N \) is the height of the mesh network. In a 2D mesh planar architecture as presented in Fig. 1, there are \( N \times (M - 1) + 2 \times M \times (N - 1) \) available full-duplex links.

Each mesh router presented in Fig. 1 can be connected to a resources tile through a network interface. Network interface (NI) is a component that packetizes data from the tile to the NoC, depacketizes packet from the NoC to the tile, and undertakes any other necessary functionality in accordance with additional communication protocol specifications. The tile can be a bus-based digital signal processor.
B. Planar Adaptive Routing Function

The description of the planar adaptive routing algorithm has been presented in our previous work [17]. If a message injected from $(x_{source}, y_{source})$ will be sent to a target node $(x_{target}, y_{target})$, and the $x$-distance between source and target nodes $(x_{offs} = x_{target} − x_{source})$ is positive or zero, then packets will be routed through the physical channels of the $X+$ subnetwork. In contrast, if $x_{offs}$ is zero or negative, then the packets will be routed through the physical channels of the $X−$ subnetwork. The ports connected to vertical links of $X+$ and $X−$ subnetworks are denoted by (North1, South1) and (North2, South2) ports, respectively. Hence, a message transported via the $X+$ subnetwork can be routed adaptively to make West–North1, West–South1, North1–East and South1–East turns as well as West–East, North1–South1 and South1–North1 straightforward routing direction. While the packets communicated via the $X−$ subnetwork will have adaptivity to choose between East–North2, East–South2, North2–West and South2–West turns as well as to choose a straightforward routing direction, i.e. East–West, North2–South2 and South2–North2 [17].

The packets are routed adaptively in the network. When a packet has two options for outgoing ports, the packet will be routed to an outgoing port having more free ID slots. More detail on such routing algorithm on the planar network topology can be found in [19].

C. Router Architecture

The generic microarchitecture of the XHiNoC router is presented in Fig. 2. The router is designed with modular-oriented method, where each modular component is regularly instantiated for each input-output port. The XHiNoC in general consists of three components in incoming port i.e., FIFO queues (comprising a Best-Effort Queue (QBE) and a Guaranteed-Throughput Queue (QGT)), a Routing Engine with multiplexed data buffering (RE) and a Grant request acknowledge (G) component. In each outgoing port, there are a Multiplexer with ID-tag Management unit (MIM) and an Arbiter (A) component. In order to keep the router size small, the depth of each virtual channel is set only to 2 slots.

VI. PACKET FORMAT FOR GT AND BE MESSAGES

Fig. 3 present the packet format used in XHiNoC. The key role of the flexible connection setup is denoted by a specific format of the XHiNoC packets. The 39-bit (0 − 38th) packet consists of a header (stream request) flit followed by payload data flits. The bits $38^{th}−36^{th}$ represents flit types of the packet, and the bits $35^{th}−32^{th}$ represents the ID (Identity) tag. Table I shows the binary encoding of 8 flit types to differentiate packets used for best-effort and guaranteed-throughput service.

A data message in the XHiNoC can be associated as a single packet, where the message is divided into several flits. In other words, the message is not divided into several packets, where each packet consists of a few flits and one header containing a routing information or destination address of the packet. Hence, the terms “packet” or “message” have similar interpretation in this paper. A message in XHiNoC, even if its size is extremely large, has only one header for one unicast message. The Computing of a routing direction for each message on each router is made once when its header is routed. Next, the payload flits, including the tail flit, will track routing paths made by the header. The message may consist of a very long flit stream. By using this packet format, there will be no out-of-order problem for each message, although an adaptive routing algorithm is used to route the stream.

The source address $(X_S, Y_S, Z_S)$ and target address $(X_T, Y_T, Z_T)$ of the packet are asserted in the header flit. The Source $Z_S$ and Target $Z_T$ fields are dedicated for addressing the resource tiles of hierarchical networks (e.g. in tree-based topologies), when our NoC will be extended to be a hierarchical NoC, or are used to address the 3D locations of computing resources (tiles). The extended address can also be used, when our NoC will be designed to be a stacked 3D NoC. The sub-hierarchical networks will be connected to a local port of each mesh node. Hence, each resource tile located in the subnetwork will have $(X, Y, Z)$ address, where the $(X, Y)$ denotes the 2D address of the mesh network and the $(Z)$ represent the address of the tile in the sub-hierarchical network. Each flit, which belongs to the same message, has the same local identity number (ID-tag). The unique local ID is used to differentiate each flit from the other packets when it passes through a communication link.

VII. ROUTING SERVICES FOR DIFFERENT PACKETS

A. Guaranteed-Throughput (GT) Packets

The process to establish and to terminate connection for the guaranteed-throughput message can be described in four different phases as depicted in Fig. 4, in which core A sends a data stream to core B via the on-chip network.

1) To initiate a connection, core A sends a request flit to core B as shown in Fig. 4(a).
2) After receiving the request flit, core B will analyze the request flit to find out whether the requested connection is successful or not. core B will send then a response flit as presented in Fig. 4(b) to tell core A the connection process.
3) If the connection from core A to core B is successfully established as indicated by the response flit accepted by core A, then core A will start sending the data stream to core B as depicted in Fig. 4(c).
B. Best-Effort (BE) Packets

But, if the request flit fails to establish connection from core A to core B as indicated in the “info” field by the response flit, then core A will terminate the connection by sending a tail flit to remove the reserved communication resources as presented in Fig. 4(d). Afterwards, core A will start again to send a new request to establish connection to core B.

A requested connection can fail because there is no more available ID-slot in certain communication resources in intermediate routers. Our current router implementation uses 4 bits for ID-tag field. It means that there are 16 ID slots available on each communication resource. However, we use only 15 ID slot for communication and the remaining one ID slot is reserved to control the flow of header flits which flow in the links that run out of ID slots. In the design, we use ID-tag “1111” as the control ID-tag. For instance, if a header flows through a link that run out of ID slot, then the header will be assigned with the ID-tag “1111”. Once a header is assigned with the ID-tag “1111”, then it will be always assigned with the ID-tag “1111” on each communication link until it reaches its destination node.

B. Best-Effort (BE) Packets

Beside the GT packets, the NoC can route also the BE packets. A best-effort data communication is connectionless. In our XHiNoC, the best-effort databody and the last databody (the tail flit) are sent by following its header flit injected in advance without waiting for a response flit from the destination node. Hence, we provide a different mechanism to handle a message that cannot reserve an ID slot in a certain intermediate node. As explained in Subsection VII-A, a header entering a link which run out of ID slot will be assigned with ID-tag “1111”, and will be always assigned with the ID-tag “1111” when entering the next communication resources until it reaches the destination node.

The aforementioned rule is also valid for headers of the BE messages. But, the BE payload flits belonging to the same header having ID-tag “1111” on the considered link will be dropped in the outgoing ports of the link. This data dropping mechanism is provided for the BE messages in our NoC to avoid deadlock (due to the possible ID slot run out problem), because the BE payload data are injected soon after the header flit have been injected from source node without waiting for a response flit from target node to let the source node recognize whether the header has successfully reserved one ID slot on every required communication link connecting the source and the target node.

Furthermore, after the header flit is accepted by the destination node, then the node will send a response flit to the source node. After analyzing the response flit, the source node will stop injecting the best-effort data, send the tail flit to remove ID slot utilization in each router, and send again the message. In our current XHiNoC prototype, we have provided 15 ID slots plus 1 ID slot reserved for link over-capacity control purpose on each link. We can still increase available ID slots per link at design time. For instance, with we use 5-bit ID-tag field on each flit, then 32 ID slots will be available on each link.

VIII. ID-BASED DATA FLOW CONTROL

A. ID-based Routing Mechanism

Routing engine (RE) units in the XHiNoC combines a routing state machine (RSM) and a routing look-up table (LUT) unit. This combination is useful to support a runtime link interconnect configuration. Every flit brings also its flit type and ID-tag together with a data word. A local ID slots is defined as a set Ω = {0, 1, 2, ..., N_{id} - 1} where N_{id} is the number or maximum number of available local ID slot on each communication link of the NoC. A flit flowing on a communication link can be classified as BEHeader, GTHeader, BE databody, GT databody, BETail, GTTail, Response.

In the context of the mesh planar router structure presented in this paper, then we can assign routing direction East, North1, West, South1, North2, South2 and Local as output port direction 1, 2, 3, 4, 5, 6 and 7, respectively.

If the RE unit detects a packet header (request flit) having ID-tag F_{id} ∈ Ω from the input port, then the RSM unit will look for a correct routing direction based on destination address stated in the header flit and the current address of the router. The routing direction is then asserted in a register number F_{id} of the LUT unit (indexing based on the ID-tag F_{id} of the packet). In the next time periods as the RE unit detects payload data flits having ID-tag F_{id}, then their routing direction will be get directly from the LUT unit according to their ID-tag number indexed before. Fig. 5 shows an example of the ID-based routing mechanism in a mesh router. The packet, having ID-tag 2, is routed to the Local port and its ID-tag is updated to a new ID-tag number, i.e. 1.

B. Optimal Link Utilization

Fig. 6 shows some packets (pck A, B, C, D and E) can be interleaved each other to share the same communication link. In our NoC implementation rule, flits belonging to the same packet will have the same ID-tag on each link. Hence, the ID-tag attached in each flit enables each payload data to track their correct routing direction. In other words, the ID-tag represents the compressed form of routing direction made by a header flit. The header flit reserves communication resources by using one ID slot in an ID Slot Table at each outgoing link, and use this slot as its current ID-tag. Each flit is then routed in accordance with its current local ID-tag, where header flits find the routing direction and compress it in a Routing Table, while payload flits will extract the routing direction from the Routing Table by using its current ID-tag as the table index. Remember that flits belonging to the same message will always have the same local ID-tag on each communication link. Each interleaved flit can extract the required routing direction by searching it in the routing table in accordance with its ID-tag.
IX. EXPERIMENTAL RESULTS

This section presents the effectiveness of our methodology to combine the best-effort and connection-oriented guaranteed-throughput data delivery services. We use a transpose traffic scenario running on 4x4 mesh planar topology, where a message will be injected from source node (i,j), and will be accepted in target node (j,i). Fig. 1 presents the detail of the 2D 4x4 mesh planar addresses. In the 2D 4x4 mesh with the matrix transpose traffic, we will have 6 node communication pairs. Communication 1 (Comm1) is a communication pair between node (1,0) as data injector and (0,1) as data acceptor, and is represented in this paper as Comm1(1,0) ⇒ (0,1). Hence, the rest communication pairs can be represented as: Comm2(2,0) ⇒ (0,2), Comm3(3,0) ⇒ (0,3), Comm4(2,1) ⇒ (1,2), Comm5(3,1) ⇒ (1,3) and Comm6(3,2) ⇒ (2,3).

A traffic pattern generator (TPG) and a traffic response evaluator (TRE) is implemented on each network node. The TPG unit encodes each packet or message such that every message can be recognized and differentiated from other messages. Each flit of a packet is numbered in order by the TPG unit. The TRE unit will check every accepted flit and evaluate if any or some flits loose in the NoC or are not accepted in a destination node. The TRE unit at destination node will analyze also the header, databody and the tail flit of a packet, and check whether the accepted packet has correctly reached its destination node. The TRE unit counts also the number of clock cycles required by the header and other flits to reach the destination node. We translate the latency metric in our simulation as the number of clock cycles to transfer a flit from its source node to its destination node, where initial clock count is set at simulation start time. In other words, we do not reset initial clock count when a new flit is injected from its source node. Hence, in the simulation result figures, we will see that the transfer latency will increase linearly. The slope of the graphic represents the communication bandwidth (flit acceptance rate) of each source-to-target communication. The smaller the slope, the higher the communication bandwidth is.

In the first experiment, two independent simulations are made. The first simulation is run in which all data injector nodes send BE messages, while in the second simulation, all data injector nodes send GT messages and receive response flits from consumer nodes. This experiment is intended to present the effect of connection mechanism over transfer latency, where a data injector must wait for a response flit from a data acceptor before injecting the streaming data. 50 flits are injected from each source node in both simulation, and the flits are injected with 0.25 fpc (flit per clock cycle). In other words, a new flit is injected in every 4-clock cycle. Fig. 7 presents the comparison of tail flits transfer latency of all communication pairs. It is clear that, the transfer latencies of the GT tail flits in the first simulation are larger than the latencies of the BE tail flits in the second simulation, because the BE payload data are injected to the NoC soon after the BE header flit (no latency to wait for a response flit). The tail flit transfer latencies of the GT tail flit is approximately similar to the transfer latencies of the BE tail flit plus the transfer latency of the response flit to the data injector (measured when the data injector starts injecting the header flit).

The second experiment is run, in which BE and GT messages are mixed in the matrix transpose traffic scenario. As explained before, we have 6 communication pairs in the transpose traffic pattern. We set Comm2, Comm4 and Comm6 as GT-type injector-acceptor communication pairs, while Comm1, Comm3 and Comm5 as BE-type injector-acceptor communication pairs. The transfer latencies of each tail flit increase linearly as the number of workloads (flits per producer node) is increased. In the simulations, the workloads are increased from 250 flits per data producer until 4000 flits per producer. The injection rate per producer is also changed between 0.1, 0.125 and 0.25 fpc (flit per clock cycle), which means that a data flit is injected to the NoC in every 10, 8 and 4-clock cycle, respectively.

The simulation result shown in Fig. 8 has presented a very interesting characteristic of the XHiNoC. It performs a very flexible runtime communication resource reservation even if the data injection rates at each producer node are changed. It is certainly difficult to obtain such characteristic if we use the time-slot-based TDM method which has been used by other NoC proposals. The data communication is also lossless, i.e., all injected flits in source nodes are accepted in target nodes. We run the simulations by encoding each message such that every message can be recognized the data flits of each data communication pair and to ensure that there is no loss of data.

X. CONCLUSION AND FUTURE WORKS

This paper has presented a novel approach to combine the connectionless BE and the connection-oriented GT services in the NoC by using the runtime dynamic ID-based routing method, in which the communication resource reservation is made during application execution time (at runtime) based on the dynamic local ID slots allocation and ID-based organization technique. The proposed method does not require a specific slot allocation algorithm that must be run at compile time.

By using our runtime ID-slot-based routing and scheduling, the injection rate of each GT packet can be freely determined, or predetermined according to the required injection rate (required bandwidth communication). This interesting characteristic is difficult to achieve by using its counterpart time-slot-based scheduling in case that routing and scheduling are made at runtime during application execution. This guaranteed bandwidth service will provide a good interconnect platform for many core processor systems. Hence, applications requiring high-bandwidth interprocessor communications such computer vision and multimedia applications can be run with better performance.

REFERENCES


83
Fig. 8. Simulation with variable workloads and injection rates (fpc = flit per cycle).


