Design of a Feasible On-Chip Interconnection Network for a Chip Multiprocessor (CMP)

Seung Eun Lee, Jun Ho Bahn, and Nader Bagherzadeh
Department of Electrical Engineering and Computer Science
University of California-Irvine, Irvine, CA 92697-2625
{*seunglee, jbahn, nader*}*@uci.edu*

Abstract

In this paper, an adaptive wormhole router for a flexible on-chip interconnection network is proposed and implemented for a Chip-Multi Processor (CMP). It adopts a wormhole switching technique and its routing algorithm is livelock-/deadlock- free in 2D-mesh topology. Major contribution of this research is the design of an adaptive router architecture adopting a minimal adaptive routing algorithm with near optimal performance and feasible design complexity, satisfying the general SoC design requirements. We also investigate the optimal size of FIFO in an adaptive router with fixed priority scheme.

1 Introduction

In order to meet the design requirements for computation intensive applications and the needs for low-power, high-performance systems, the number of computing resources in a single-chip has been enormously increased, and this is mainly because current VLSI technology can support such an extensive integration of transistors. With this trend, on-chip interconnection network among multiple processors becomes another challenging issue in the current Chip-Multi Processor (CMP) design.

In this paper, an adaptive router algorithm and an architecture for a flexible on-chip interconnection are proposed and applied as an on-chip interconnection method for our CMP. Major contribution of this research is the design of an adaptive router architecture adopting a minimal adaptive routing algorithm with near optimal performance and feasible design complexity, thus satisfying general SoC design requirements. The FIFO buffer is a key component of an interconnection network. We also investigate the performance characteristics of FIFO for an on-chip interconnection network and propose an optimal size of FIFO to increase throughput and to reduce physical cost in the context of an on-chip network.

The rest of the paper is organized as follows: Section 2 presents the motivation for this research. The architecture of the proposed router is described in Section 3. Section 4 presents experimental results of the proposed router. Finally, conclusions are drawn in Section 5.

2 Motivation

As a new SoC design paradigm, Network-on-Chip (NoC) [6, 20] has been proposed to support the trend for SoC integration. The basic idea is the use of packet switching methodology that has been extensively used for computer network. Even though the basic idea of NoC comes from the computer network and the network technology in computer network is already well developed, it is almost impossible to apply them to a chip level interconnection environment without any modification or SoC optimizations. To be practical for an NoC architecture, the basic functionality should be simple and light-weighted. Consequently, implemented components of an NoC architecture would be small enough to be suitable for construction on future SoCs. On the other hand, it must fulfill the communication requirements on application systems.

Recently, a number of research projects have shown how to develop network architecture to be appropriate for on-chip environments. Different routing algorithms, such as deterministic/oblivious and adaptive routing algorithms, have been proposed. Because of simplicity and ease of analysis, many researchers adopted deterministic/oblivious algorithms such as DOR [19], ROMM [16], and O1TURN [18]. Adaptive routing algorithms for better performance than deterministic/ oblivious ones have been proposed [11, 5, 7, 3, 9, 17]. Recently there have been some implementation related works using deterministic routing algorithms as well as some adaptive routing ones [10, 14, 13].

The design of a high performance router for on-chip interconnection has tight resource constraints such as router's

1550-6533/07 $25.00 © 2007 IEEE
DOI 10.1109/SBAC-PAD.2007.38

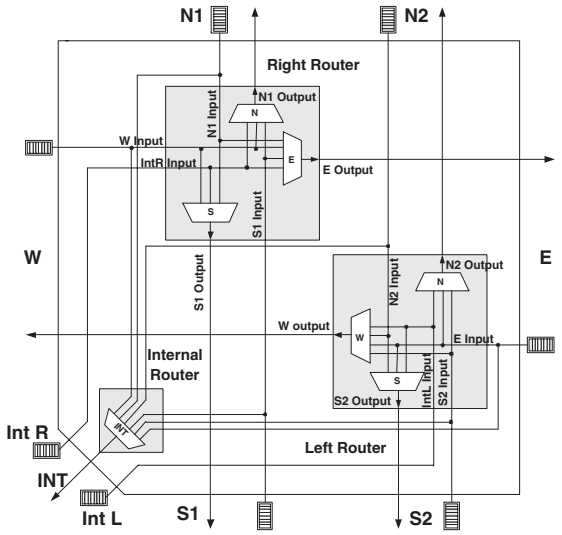

Figure 1. Adaptive wormhole router architecture

area, power, and speed. Our goal is to provide a feasible router architecture for on-chip realization by evaluating design trade-offs. The adaptive routing has been proposed as a method of improving network utilization by using information about the network state to select a path among alternative paths to deliver a packet. And the wormhole flow control has increasingly been advocated as a means of reducing latency. It reduces latency by routing a packet as soon as its head flit arrives at a node. The routing of a head flit enables the switch to establish the path and body flits are simply forwarded along the path as a pipelined fashion.

In this paper, we will propose the adaptive wormhole router to realize on-chip interconnection network for CMP. We will also show the performance of the proposed router and demonstrate the viability of our proposal by providing implementation results.

3 Adaptive Wormhole Router Architecture

3.1 Overview

Figure 1 illustrates a single router with 8 input and 7 output channel ports, employed in every node of the network. There are two north and two south channel pairs reaching each router, providing two disjoint sub-networks for the west-to-east and east-to-west traffics using the channel sets $\{$W-in, N1, E-out, S1$\}$ and $\{$E-in, N2, W-out, S2$\}$, respectively. This separation of routing paths for vertical direction and unidirectional path for horizontal direction allow a deadlock-free operation [5, 7, 15]. Also by choosing a shortest path in routing, a livelock free operation is guaranteed [4, 8].

The router supports four kinds of control/data transfer packets: (1) *SINGLE* data transfer packet, (2) *SINGLE* command transfer packet, (3) *BLOCK* program transfer packet and (4) *BLOCK* data transfer packet. The head flit has a necessary information for routing such as *destPE_addr* field to indicate the destination processing element (PE). The address of destination PE is represented in signed magnitude value of the relative distance of horizontal and vertical direction, i.e. MSB of each *X-dir* and *Y-dir* field represent its sign and the rest of bits represent its magnitude. A positive value represents southern and eastern direction in vertical and horizontal direction, respectively. Similarly a negative value represents northern and western direction.

For the outgoing channel allocation, the router applies a fixed priority scheme to reduce the complexity of the router. The possible incoming channels have a descending order of priority in a clock-wise direction for each outgoing channel. Similarly, for the outgoing channels, a clock-wise order of priority starting from N1 is given. The incoming packets from PE and output channel to PE have the lowest priority in each priority group. Additionally, the *BLOCK* transfer packet has a higher priority than the *SINGLE* transfer packet among different types of packet.

The routing algorithm was compared with DOR [19], ROMM [16], and O1TURN [18] algorithms in [2]. The router models were written in SystemC and simulations were executed with different traffic patterns. Our routing algorithm showed same or better performance for all traffic patterns in 4×4 mesh topology. For every traffic pattern, it sustained highest offered traffic amount with the lowest average latency. Though ours had slightly lower performance than O1TURN at 8×8 mesh topology, it still showed competitive performance. However, at the given amount of offered traffic before saturation point, it demonstrated the best performance with respect to the average latency.

3.2 Design of the Adaptive Wormhole Router

There is an input FIFO queue per each input channel and every output port has an associated arbiter (see Fig. 1). We consider a router with seven interfaces, suitable for a 2D mesh with an additional interface to a PE. We assume that a packet coming through an input port does not loop back, thus each input port is connected to the corresponding output ports except for itself in the associated channel set.

The proposed router is composed of three architectural blocks named: *Right Router*, *Left Router*, and *Internal Router*. The *Right Router* serves the channel set $\{$W-in, N1, E-out, S1$\}$ and the *Left Router* serves the channel set $\{$E-in, N2, W-out, S2$\}$. The *Internal Router* supports the

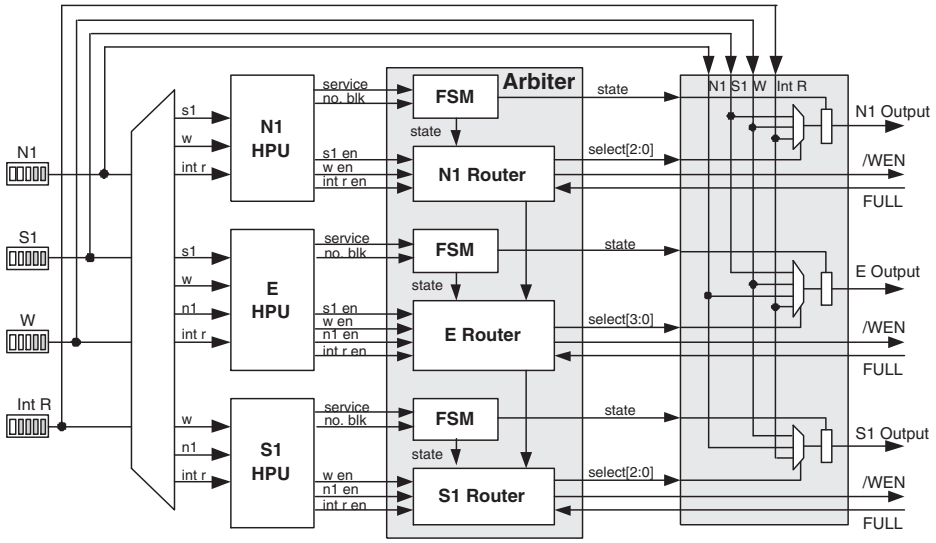

Figure 2. Block diagram of the *Right/Left Router*

Table 1. Signal definition for routing algorithm description

Signal Name	Description
U	set of incoming packet
u	incoming packet
v	output port
$u.direction$	direction of incoming packet and output port
$v.u.en$	incoming packet u can be routed to output port v
$u.service$	service type for incoming packet
$u.no\_blk$	number of body flits in incoming packet
$u.empty$	FIFO empty signal for incoming port
$v.state$	state of the FSM for output port
$v.remained\_no\_data$	number of remained body flits for *BLOCK* service
$v.select$	routed incoming packet to output port
$v.full$	FIFO full signal for output port

additional interface to a PE. The detail block diagram of the *Right Router* is shown in Figure 2 and Table 1 summarizes the signal definition in the following description of routing algorithm.

The routers for each output port are placed according to the priority of outgoing channel from the uppermost router (N1) to the lowest router (S1). Each incoming packet is directed to the header parsing unit (HPU) per each output port. The HPU generates a set of possible incoming packets which could be routed to the corresponding output port, in accordance with the input priority level by looking up the destination address in the header field.

Algorithm 1 HPU(U,v)

for each u **do**
  if $u.direction = v.direction$, $v.full$ =FALSE **then**
    $v.u.en \leftarrow$ TRUE
  else
    $v.u.en \leftarrow$ FALSE
  end if
  if $u.service$ =BLOCK **then**
    $v.service \leftarrow$ BLOCK
  else
    $v.service \leftarrow$ SINGLE
  end if
end for

Our router supports two service levels: *BLOCK* transfer mode and *SINGLE* transfer mode. If one output port is granted to an incoming packet with the *BLOCK* transfer mode, it reserves the output port until all flits in a packet are forwarded. To support this feature, our router has a finite state machine (FSM) in each output port, which serves dual transfer modes by storing service state (*SINGLE/BLOCK*). Also it records the remaining number of body flits that would be forwarded in each time. In *BLOCK* transfer mode, the body flits have their own path in order to forward a packet to the selected output port reserved by head flit. Thus the body flits arriving at the input channels can continue advancing along the reserved output ports without passing through the arbiter unit.

Algorithm 2 FSM(U,v)

```
for each u in the given priority by output port v do
  if v.state = BLOCK then
    if v.remained_no_data > 0 then
      if u.empty = FALSE then
        v.remained_no_data − −
      end if
      v.state ←BLOCK
    else
      v.state ←SINGLE
    end if
  else
    if v.select = u, u.service =BLOCK, u.empty =FALSE
    then
      v.state ←BLOCK
      v.remained_no_data ← u.no_blk
    else
      v.state ←SINGLE
    end if
  end if
end for
```

The arbiter gives a way for competing incoming packets to corresponding output ports according to their priority levels. After completing the path decision of the first router (N1), the second router (E) refers to the path decision of the first router and disables the incoming port, which was served by the first router, from the set of possible incoming packets for the second router. The third router (S1) completes the routing for the set as the same as the second router does. In this way, the arbitration is done by propagating the path decisions from the uppermost router to the lowest router. Therefore all incoming packets could be forwarded in their output port at once.

The arbiter generates *select* signal for each output port. The multiplexer maps the corresponding incoming packet to the output port by referring to the *select* signal. Finally, it updates the destination address in the head flit according to the service mode in order to complete proper transmission. In *SINGLE* transfer mode, it decreases the corresponding destination address. In *BLOCK* transfer mode, it decrease the corresponding destination address in the head flit and bypasses the body flits.

Algorithm 3 ARBITER

```
S ← 0
for each input port u do
  for each possible output port v do
    if v.u.en = TRUE then
      S ← v.u.en
    end if
  end for
end for
for each output port v in the given by priority do
  if v.state = BLOCK then
    v.select ← previous v.select
  else
    for each input u in the given priority by output port
    do
      if v.u.en ∈ S then
        v.select ← u
        remove all.u.en from S
      end if
    end for
  end if
end for
```

The *Left Router* is the symmetric version of the *Right Router*. Similarly, the *Internal Router* consists of an HPU, an FSM, and an arbiter for the incoming packets to the internal node.

In order to achieve high performance, all routing decisions are made within one clock cycle. So, an incoming flit can advance in one clock cycle concurrently when the corresponding output port is free.

3.3 Physical Implementation

In order to show the router's performance and its feasibility for VLSI implementation, a logic description of our router was obtained using the *Synopsys*^{TM} $V-2003.12$ with *TSMC*^{TM} $90nm$ technology. Table 2 summarizes the physical characteristics of a router and a FIFO.

The *Synopsys*^{TM} tool chain provided critical path information for logic within the single router up to $423MHz$. The operating frequency of the router is less than static routers since our switching mechanism selects adaptive routing strategy and supports multiple service levels. The main drawback of an adaptive router is the increase in the critical path according to the number of competing channels and the number of service levels. The arbiter should take

Table 2. The physical characteristics of an adaptive router and FIFO

	Router	FIFO (Depth 8)
Voltage	1.0 V	1.0 V
Frequency	423 MHz	1.8 GHz
Area	17,537 μm^2	17,428 μm^2
Dynamic power	2.47 mW	10.12 mW
Leakage Power	0.17 mW	0.16 mW

care of the service levels and the priorities of incoming and outgoing ports, and then it decides a routing path. However, an adaptive routing can balance network occupancy and enhance its maximum throughput. In SoC layout, wire lengths are very non-uniform due to the irregular floor plan. While the wire length can be very long leading to timing and clocking issues, our system assumes regular mesh network and well segmented layout of PEs. Therefore, this physical characteristic of the single router is well suited for CMP.

The performance of the router in terms of average latency depends on the FIFO size. Our results show that FIFO with depth of 8 is optimal when the packet size is 8. The overall router including the input FIFOs with size 8 occupies an area of approximately $0.157mm^2$ ($Router\ Area + FIFO\ Area \times 8$) using the $90nm$ technology. The *ARM11 MPCore*^{TM} and *PowerPC*^{TM} *E405*, that provide multi CPU designs, occupy $1.8mm^2$ and $2.0mm^2$ in $90nm$ technology, respectively [1, 12]. If the router and FIFO were integrated within a CMP as an interconnection network, the area overhead imposed by the network would be reasonable showing the feasibility of this approach.

4 Experimental Results

4.1 Evaluation Methodology

The performance of an interconnection network can be described by latency versus offered traffic curves. Although latency versus offered traffic curves give the most accurate view of the ultimate performance of an interconnection network, they do not have simple, closed form expressions and are generally found by discrete event simulation [4]. For the measurement of throughput and adjusting incoming traffic, we adopted a standard interconnection network [4] where the packet generation is placed in front of an infinite depth source queue and input timing of each packet is measured whenever it is generated.

Recently, several specific communication patterns between pairs of nodes have been used to evaluate the performance of interconnection networks. These communication patterns take into account permutations that are usually performed in parallel numerical algorithms [4, 8]. In this paper, simulation is completed using four different traffic patterns such as uniform random, matrix transpose, bit-complement, and bit-reverse traffics in 4×4, and 8×8 mesh networks varying the size of an associated FIFO. Even though these traffic patterns can not realistically reflect the type of real traffic that will traverse the network in CMP, they are generally used to evaluate the performance of a network.

4.2 Simulation Results

In this simulation, the packet length is fixed to 8 flits (1 head flit and 7 body flits) even though the packet format supports various sized packets. Each graph represents offered traffic (flit/node/cycle) in *X*-axis and average latency (cycles) in *Y*-axis.

For the 4×4 network (see Fig. 3), the adaptive router makes the phase transition region shift from 0.24 flits/cycle to 0.28 flits/cycle with uniform random traffic. In case of matrix transpose traffic, the result shows similar performance except for the smaller FIFO depth than message length (FIFO depth of 4). With the bit-reverse and the bit complement traffic, the average latencies are decreased with smaller FIFO size except for the FIFO depth of 4. For the 8×8 network (see Fig. 4), overall performance of the router with four traffic patterns show similar tendency as with 4×4 network. The only difference is reduction of the throughput. The experimental results demonstrate that the effective bandwidth of the interconnection network is up to $6.63Gbps$ and $2.84Gbps$ per each PE in 4×4 and 8×8 mesh network, respectively.

Regarding the FIFO allocation, experimental results show that increase of the FIFO size does not always improve the average latency. Router shows the best performance when FIFO size is 8 (the same as packet length) with all other traffic patterns except the uniform random traffic. In order to reduce the complexity of the router, we adopted a fixed priority scheme for outgoing channel allocation. In case of matrix transpose, bit-reverse, and bit complement traffic patterns, there is a chance of having hot-spot nodes that can impact overall latencies. Even though the adaptive routing algorithm has an ability to balance the load across the channel, the alternative path decision can be made under congestion situation. If receiver has enough capacity, sender would forward the packet to the receiver without choosing an alternative path reducing channel utilization. Therefore, a larger FIFO allocation for such a network does not improve router performance. In other words, larger sized FIFO can accumulate the incoming packet in the FIFO increasing average latency. For that reason, the performance using FIFO depth of 8 shows better performance than using FIFO depth of 64 in matrix transpose, bit-reverse, and bit-complement traffics. *Thus, we can conclude that allocating*

FIFO depth with the same number of flits in a packet has optimal performance in terms of average latency and physical cost in an adaptive router using fixed priority scheme.

5 Conclusions

In this paper, an adaptive router architecture for a flexible on-chip interconnection is proposed and implemented. We also showed that the optimal FIFO size for an adaptive router which uses fixed priority scheme is the same as the packet length. Moreover, the proposed router was synthesized using $90nm$ CMOS technology. The performance and the total area overhead of the router demonstrated the feasibility of the on-chip interconnection network for a CMP realization.

References

[1] ARM. Arm11 mpcore. http://www.arm.com.

[2] J. H. Bahn, S. E. Lee, and N. Bagherzadeh. On design and analysis of a feasible network-on-chip (noc) architecture. In *Proceedings of International Conference on Information Technology (ITNG'07)*, pages 1033–1038, 2007.

[3] G.-M. Chiu. The odd-even turn model for adaptive routing. *IEEE Trans. Parallel Distrib. Syst.*, 11(7):729–738, 2000.

[4] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., 2003.

[5] W. J. Dally and C. L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Comput.*, 36(5):547–553, 1987.

[6] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *Design Automation Conference*, pages 684–689, 2001.

[7] J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Trans. Parallel Distrib. Syst.*, 4(12):1320–1331, 1993.

[8] J. Duato, S. Yalamanchili, and L. Ni. *Interconnection Networks: An Engineering Approach*. IEEE Computer Society Press, 1997.

[9] C. J. Glass and L. M. Ni. The turn model for adaptive routing. *J. ACM*, 41(5):874–902, 1994.

[10] K. Goossens, J. Dielissen, and A. Radulescu. Athereal network on chip: Concepts, architectures, and implementations. *IEEE Des. Test*, 22(5):414–421, 2005.

[11] J. Hu and R. Marculescu. Dyad: smart routing for networks-on-chip. In *DAC '04: Proceedings of the 41st annual conference on Design automation*, pages 260–263, 2004.

[12] IBM. Ibm powerpc 405 embedded core. http://www.ibm.com.

[13] S. E. Lee and N. Bagherzadeh. Increasing the throughput of an adaptive router in network-on-chip (noc). In *CODES+ISSS '06: Proceedings of the 4th international conference on Hardware/software codesign and system synthesis*, pages 82–87, 2006.

[14] S.-J. Lee, K. Lee, and H.-J. Yoo. Analysis and implementation of practical, cost-effective networks on chips. *IEEE Des. Test*, 22(5):422–433, 2005.

[15] X. Lin, P. K. McKinley, and L. M. Ni. The message flow model for routing in wormhole-routed networks. *IEEE Trans. Parallel Distrib. Syst.*, 6(7):755–760, 1995.

[16] T. Nesson and S. L. Johnsson. Romm routing on mesh and torus networks. In *SPAA '95: Proceedings of the seventh annual ACM symposium on Parallel algorithms and architectures*, pages 275–287, 1995.

[17] V. Puente, R. Beivide, J. A. Gregorio, J. M. Prellezo, J. Duato, and C. Izu. Adaptive bubble router: A design to improve performance in torus networks. In *ICPP '99: Proceedings of the 1999 International Conference on Parallel Processing*, pages 58–67, 1999.

[18] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi. Near-optimal worst-case throughput routing for two-dimensional mesh networks. In *ISCA '05: Proceedings of the 32nd Annual International Symposium on Computer Architecture*, pages 432–443, 2005.

[19] H. Sullivan and T. R. Bashkow. A large scale, homogeneous, fully distributed parallel machine, i. In *ISCA '77: Proceedings of the 4th annual symposium on Computer architecture*, pages 105–117, 1977.

[20] N. Tabrizi, N. Bagherzadeh, A. H. Kamalizad, and H. Du. Mars: a macro-pipelined reconfigurable system. In *CF '04: Proceedings of the 1st conference on Computing frontiers*, pages 343–349, New York, NY, USA, 2004. ACM Press.

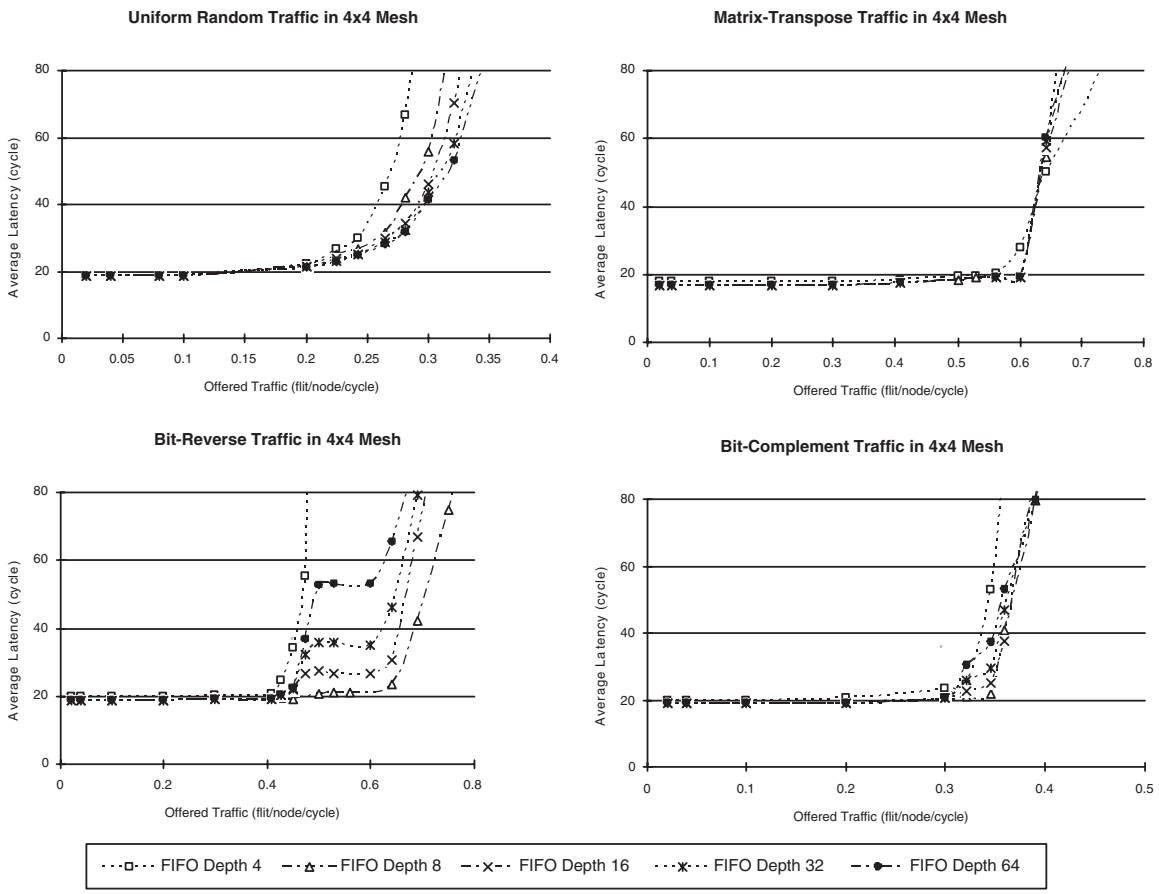

Figure 3. Average latency in 4×4 with varying FIFO size

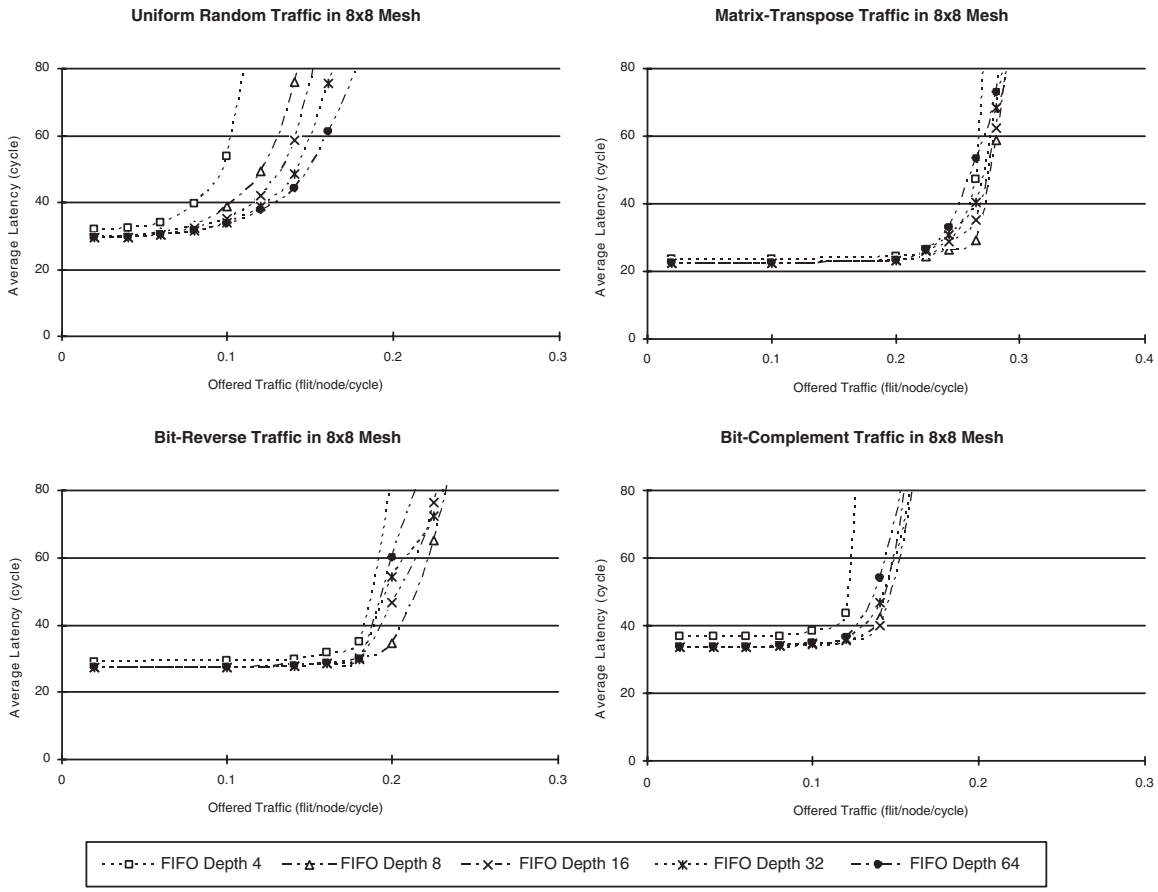

Figure 4. Average latency in 8×8 with varying FIFO size