Design of Adaptive Communication Channel Buffers for Low-Power Area-Efficient Network-on-Chip Architecture

Avinash Kodi[†], [*] Ashwini Sarathy[‡] and Ahmed Louri[‡]
[†] Dept. of Electrical Engineering and Computer Science, Ohio University, Athens, OH 45701
[‡] Dept. of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721
{avinashk, sarathya, louri}@ece.arizona.edu

ABSTRACT

Network-on-Chip (NoC) architectures provide a scalable solution to the wire delay constraints in deep submicron VLSI designs. Recent research into the optimization of NoC architectures has shown that the design of buffers in the NoC routers influences the power consumption, area overhead and performance of the entire network. In this paper, we propose a low-power area-efficient NoC architecture by reducing the number of router buffers. As a reduction in the number of buffers degrades the network's performance, we propose to use the existing repeaters along the inter-router links as adaptive channel buffers for storing data when required. We evaluate the proposed adaptive communication channel buffers under static and dynamic buffer allocation in 8×8 mesh and folded torus network topologies. Simulation results show that reducing the router buffer size in half and using the adaptive channel buffers reduces the buffer power by 40-52% and leads to a 17-20% savings in overall network power with a 50% reduction in router area. The design with dynamic buffer allocation shows a marginal 1-5% drop in performance, while static buffer allocation shows a 10-20% drop in performance, for various traffic patterns.

Categories and Subject Descriptors

B.4.3 [**Hardware**]: Input/Output and Data Communications—*Interconnections (Subsystems)*; C.5.4 [**Computer Systems Organization**]: Computer System Implementation—*VLSI Systems*

General Terms

Design, Performance

Keywords

Network-on-Chip, Low-Power design

*This work was performed at the High-Performance Computing Architecture and Technologies (HPCAT) Laboratory, University of Arizona, Tucson.

ANCS'07, December 3–4, 2007, Orlando, Florida, USA.

1. INTRODUCTION

As feature sizes decrease to the deep sub-micron regime, the trend towards integrating more functionality onto a single chip has led to the rise of the System-on-Chip (SoC) paradigm [1, 2]. In SoC architectures, gate delays continue to scale down with successive technology generations while interconnect delays increase [3]. This increased wire delay constraint in SoCs has driven the design and development of a modular and scalable packet-switched Network-on-Chip (NoC) paradigm [1, 2, 4, 5, 6, 7, 8]. As NoCs are being targeted at complex systems such as SoCs, accurate estimation of their performance, power dissipation and area overhead are essential during the design phase. These on-chip networks are characterized by the *channels* for data transmission and the *routers* for storing, arbitration and switching functions performed by input buffers, arbiters and the crossbar respectively. In a recent workshop on on-chip interconnection networks [9], it was shown that almost 46% of the router power was consumed by the input buffers and 54% of the router area was dominated by the crossbar. The increasing need for low-power NoC architectures has initiated several research efforts into optimizing the buffer design [5, 6], minimizing the crossbar power [4, 7] and improving the network performance [4, 10, 11, 12].

It is well known that the input buffers account for significant router power budget and chip area. A straight forward optimization would be to reduce the number of input buffers, which would reduce both the power consumption as well as the area overhead. However, the performance and flow control of the interconnection network is primarily characterized by the input buffers [13]. A good flow control determines how a network's resources, such as the channel bandwidth and the buffer capacity are allocated to packets traversing the network [13]. As opposed to store-and-forward switching which required a large buffer capacity, wormhole switching [13] allowed the buffer capacity to be reduced, such that the flits (the basic unit of flow control, a packet consists of several flits) of the same packet could potentially be in several routers. Though wormhole switching alleviated the need for larger buffers, the channel state was coupled to the channel bandwidth which in turn caused head-of-line (HoL) blocking. Virtual channel (VC) flow control [13] mechanisms provided the necessary decoupling, such that the input buffer was organized as several independent flit buffers allocated to different packets. This permitted two significant improvements, (1) the throughput of the network was greatly improved as blocked packets could make progress and (2) the network could avoid po-

tential deadlock situations. For power and area constrained NoC design, reducing the size of input buffer will potentially cause a reduction in either the number of VCs or the buffer depth, both of which are very critical for overall network performance. At low network load, reduction of either VCs or the depth may not cause significant loss in performance, though at high loads, this could lead to degradation in performance from lack of sufficient buffers.

Current wire design trends have shown that signal delay along a wire increases quadratically with the length of the wire [14]. Repeater insertion along the wire makes the delay linearly dependent on the wire length [3, 8] and is required to meet the stringent timing requirements in the current high speed VLSI designs. Research initiatives into optimizing the performance of these repeaters have shown that the repeaters can also be designed to sample and hold data values when required, providing storage in addition to their conventional functionality [15]. Therefore, with repeaters as potential buffer elements, we can use them as buffers along the channel at high network loads when there are no more VCs or buffer slots in the router.

In this paper, we propose a low-power NoC design using circuit and architectural techniques at the channel and the router buffer respectively. At the channel, we deploy circuit level enhancements to the existing repeaters so that they can double as buffers when required. We provide a new circuit technique for the control block along the congestion line, to control the functionality of the repeaters. The control block enables the repeaters to adaptively function as buffers during congestion. At the router buffer, we deploy architectural techniques such as static and dynamic buffer allocation to prevent performance degradation, while sustaining or improving the performance of a generic router. When a buffer is statically assigned to a given packet, the remaining buffer slots may not be used by flits of other packets to prevent mixing of packets [6]. Although unused buffer slots may exist, the incoming packet blocks due to lack of storage leading to a decrease in network performance. Unlike static allocation, dynamic buffer allocation reserves buffer space on a per-flit basis. As each incoming flit can be allocated to any free buffer slot, this leads to higher network throughput, although at the cost of higher control management. In both static and dynamic buffer allocation schemes used in our design, the changes made pertain only to the input buffer and the allocation of the input buffer space to the incoming flits. As opposed to other NoC designs where performance is improved at the cost of major changes to the entire router architecture, we minimize the need for re-design and make necessary changes only at the input buffer.

This combination of circuit and architectural techniques using adaptive communication channel buffers and router buffers (with static and dynamic buffer allocation) allows us the flexibility to reduce the number of router buffers without significantly degrading the throughput and latency of the network. Power and area estimations in the 90nm technology node showed a power reduction of 40-50% and an area reduction of 50% when half the input buffers were removed. Cycle accurate network simulation on 8×8 mesh and folded torus networks at 500 Mhz and 1.0 V showed a marginal loss of 1-3% in throughput for dynamic buffer allocation and a drop of 10-20% for static buffer allocation. The combination of dynamic buffer allocation along with the adaptive channel buffers achieves significant power and chip area savings in the router architecture, while minimizing the reduced buffer size's impact on the overall performance of the network.

2. RELATED WORK

As the input buffers within the routers affect the overall performance of interconnection networks, several research initiatives have targeted improved buffer utilization as well as buffer size reduction. Given that the depth of the buffers per VC is an important resource in the NoC environment, an application-specific buffer management scheme that allocates the buffer depth to the VCs depending on the traffic pattern, has been explored in [5]. Dynamic buffer allocation has been explored by using link lists, circular buffers and a table-based approach [6, 13, 16, 17]. Dynamically Allocated Multi-Queue (DAMQ) [17] buffers made use of link lists by fixing the number of VCs for each input port. Fully Connected Circular Buffers (FC-CB) [16] avoided the delay caused in the link list approach and used registers which selectively shift some flits within the buffer. However, the FC-CB approach adds a delay due to the shifting. In a more non-shifting approach such as the ViChaR [6] the number of VCs and the depth of buffers per VC are dynamically adjusted based on the traffic load. The dynamic adjustment of VCs and buffers complicates the logic for the VC arbitration, switch allocation, credit return and slot tracker. In addition, increasing the number of VCs arbitrarily also increases the packet latency due to higher interleaving of the packets [13]. Therefore, in the proposed work, we adopt a dynamic VC table based approach with fixed number of VCs, thereby achieving the flexibility of storing flit buffers dynamically without excessive control overhead.

3. DESIGN OF CHANNEL BUFFERS

3.1 Channel Buffer Implementation

In this section we detail the implementation of the channel buffers and the associated control logic. The conventional repeaters along a link are sized and spaced according to the first-order RC wire delay model described in [3]. Figure 1(a) shows the interconnect with the conventional repeaters replaced by the three-state repeaters [15]. While we have adopted the design of the three-state repeaters from [15], we significantly differ in the implementation of the control block as will be explained in the next subsection. A single stage of the three-state repeaters comprises of a three-state repeater inserted segment along all the wires in the link. Each such repeater stage receives a control input from the corresponding control block. When the control input to a repeater stage is high, the repeaters in that stage function as channel buffers. When the control input to the repeater stage is low, the three-state repeaters function like the conventional repeaters. Figure 1(b) shows that in the absence of congestion, the control logic is turned 'OFF' (only the highlighted portion of the circuit is active). In the absence of congestion, the three-state repeater is equivalent to the conventional repeater. Data moves through the link without being held by the three-state repeaters. When the control block is turned 'ON' by the incoming congestion signal, the control signal tri-states the repeaters. The repeaters then function as channel buffers and the data bits are held in position. Once congestion is alleviated, the control logic turns

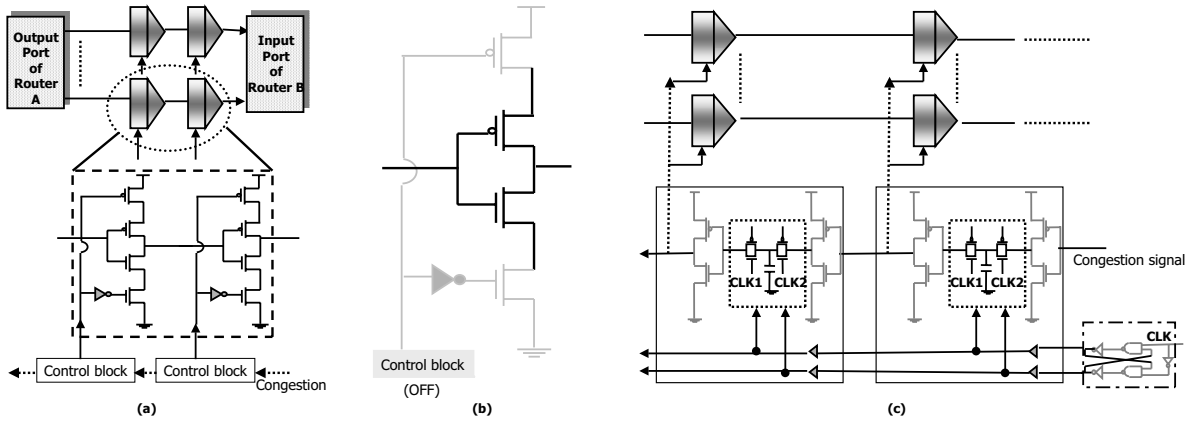


Figure 1: (a) A link using three-state repeaters that function as channel buffers during congestion. (b) Three-state repeaters functioning as conventional repeaters in the absence of congestion (c) Control blocks interfaced to the channel buffer stages.

'OFF' and the three-state repeaters function as shown in Figure 1(b). The presence of channel buffers can thus reduce the number of input buffers required in the router to achieve a given network performance. Storing the data in the channel buffers along the links is highly advantageous in terms of power and area compared to using the SRAM-based input buffers in the router. The power and area estimation are detailed in Section 5.

3.2 Control Block Implementation

The control block enables the three-state repeaters to function as channel buffers during congestion. Each channel buffer stage along the link has a corresponding control block providing the control input. Figure 1(c) shows the circuit-level implementation of the control block. The incoming congestion signal is delayed by one clock cycle at each control block, using a simple switched capacitor. In the next clock cycle, the channel buffer stage is tri-stated and the congestion signal travels to the next control block. Hence each channel buffer stage is successively turned 'OFF', to hold the data in position, until the congestion-release signal arrives. The design of the congestion control line with the control blocks shown in Figure 1(c) is more efficient than the design using conventional repeaters along the congestion control line [15] as the control block provides the following advantages: (1) The control circuit behaves as a delay module as well as a repeater for the congestion signal. Unlike conventional repeaters, the control circuit shown in Figure 1(c) operates accurately at variable clock speeds and retains the signal stability even at high clock speeds. (2) The control block can be turned 'OFF' by the clocking circuitry when there is no congestion, thus reducing the power consumption along the congestion control line.

Figure 2 illustrates the data-flow control along the channel using four stages of channel buffers and the corresponding control blocks. For simplicity only a single wire in the link has been shown. The data flow is indicated in the figure by the dark line. The congestion signal and the congestion-release signal arriving from the downstream router are indicated by the lighter solid line and the dotted line respectively. During cycle 0, there is no congestion and the channel buffers function like conventional repeaters. In cycle 1, the incoming congestion signal causes the data bit to be held by the zeroth channel buffer, while the remaining stages continue to function as repeaters. After the delay in the control block, the congestion signal travels to the next stage in cycle 2 and causes it to hold the data bit in position. The remaining two stages still continue to function as repeaters. Cycle 3 shows the congestion-release signal arriving at the zeroth stage. This causes the data in that stage to be output while the congestion signal travels to the second stage and causes it to hold the data. In cycle 4, the congestion-release signal is forwarded to the first stage and causes it to output the data held. The zeroth stage now functions as a repeater, while the congestion signal reaches the third stage, causing it to buffer the data. In cycle 5, the congestion-release signal reaches the second stage, and data is output from the second stage. The first two stages now function as repeaters. The congestion signal travels onward to the next block and the third stage waits for the congestion-release signal. Thus the channel buffers are successively switched to function as buffers during congestion, and then successively released to continue as repeaters once congestion is alleviated.

4. DESIGN OF ROUTER BUFFER

4.1 NoC Router Architecture

In packet-switched NoCs, every processing element (PE) is connected to a NoC component (router), with most NoCs commonly adopting network topologies such as mesh, or torus for regularity and modularity as shown in Figure 3(a). Current NoCs employ wormhole switching (WS) with virtual channels (VCs) per output port and adopt either deterministic (dimension-order) routing or minimal adaptive routing or both [13, 16]. In WS, each packet (comprising of several smaller, fixed-allocation units called flits) that arrives on the input port progresses through router pipeline stages before it is delivered to the appropriate output port [13]. At each intermediate router, only the header flit of every packet is responsible for the first two pipeline stages, (1) routing computation (RC) - to determine the output port and (2) virtual channel allocation (VA) - exclusive access to output virtual channel associated with the output port. Every flit (includ-

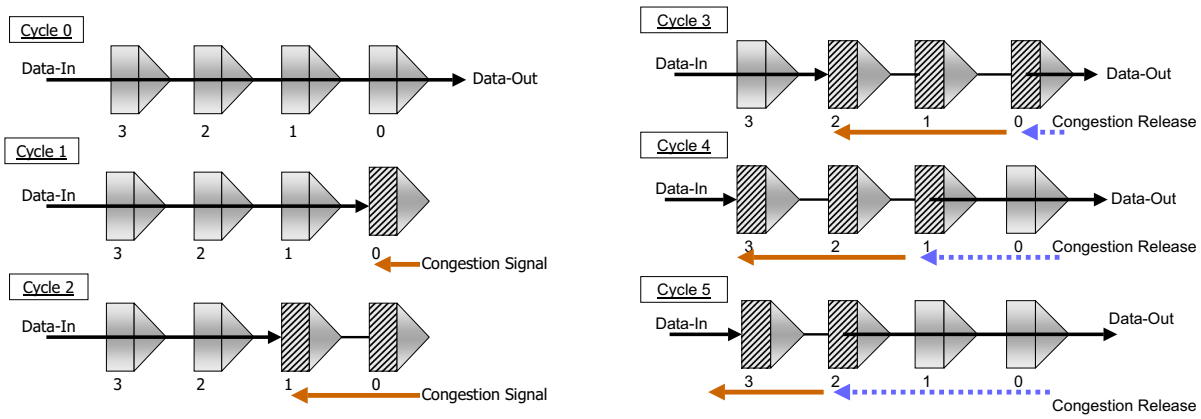


Figure 2: Data-flow control in the channel buffers during congestion.

ing header and body flits) of the packet competes for access to the crossbar in the switch allocation (SA) stage and if the flit wins the arbitration it is transferred from the input to the output port in the switch traversal (ST) stage. Each router pipeline stage requires a single clock cycle for every operation. After ST, the flit is transferred on the channel between the routers in the Link Traversal (LT) stage and the process repeats. Credit-based flow control is employed to prevent overflow of buffers, enabling reliable and in-order delivery of flits from source to destination [13]. For a router architecture with P ports, v VCs/port and r flit buffers/VC the total number of buffers/port is $z = vr$. In the VA phase, the first stage arbitration logic arbitrates among the v requests from each of the input ports, and the second stage arbitration logic arbitrates the number of VCs per input port [12]. In the SA stage arbitration [6], v:1 arbitration is used to determine a single winner per input port and P:1 output arbitration is used to determine the winner for the output port.

The input buffer is organized as a parallel FIFO with each input VC consisting of r flit buffers. Each input VC is associated with a VC state table [6, 13] that maintains the state for each incoming packet and ensures that the body flits are routed to the correct output port. The VC state table for the generic router is similar to the one shown in Figure 3(b), but does not have the C^* field. The VCID of the incoming flit allows the DEMUX to switch to the correct input VC. The RP (read pointer) points to the next flit to be transmitted and the WP (write pointer) points to a null pointer, indicating an empty flit to write the incoming data. OP (output port) is provided by the RC stage, OVC (output VC) is provided by the VA stage. CR (credits) indicates the total amount of storage availabile at the downstream router. Given that each VC has r credits, for every flit transmitted to the downstream router, a credit is consumed. Status field at the end indicates the current status of the VC - idle, waiting, routing, VA, SA, ST, and others. When the RP reads a flit out of the buffer, a credit is returned to the upstream router to indicate that it can send another flit.

4.2 Static Allocated Router Buffers

In the generic NoC design, the total number of input buffers is vr per input port. With the wires doubling as buffers, we have additional c buffers in the channel. Therefore, the total storage now available becomes $vr + c$. The number of credits available at each VC is $(vr+c)/z$. This allows routers to send additional flits into the network, even if the storage is in the channel, instead of in the router buffer. The statically allocated router buffer design with congestion control is shown in Figure 3(b). Other than the congestion control unit, all other functionalities are identical to the generic router architecture. Every VC state table maintains an additional field C^* which indicates congestion. As the buffer implemented is a FIFO buffer, if the WP does not point to a null buffer, and WP = RP, then the C^* field is set. This causes the congestion control to be activated which in turn holds the data in the network channel itself. When a flit is read from the buffer, the RP moves to the next buffer, clears congestion C^* field, which in turn allows data flits to enter into the router.

From the implementation perspective, this nominal change does not impact the design of the network router architecture. Moreover, significant power savings and area gain can be obtained. However, from the perspective of performance, this design leads to head-of-line (HoL) blocking in the channel buffers at high network loads. When the congestion field C^* is set for a particular VC, the corresponding flits are held in the network channel and block the flits headed towards other VCs, although the other VCs may have their C^* field cleared. Therefore a more attractive alternative is dynamic allocated router buffers explained in the next section.

4.3 Dynamic Allocated Router Buffers

In designing dynamic allocated router buffers, our goal is to maximize the throughput of the network as compared to the generic case. At the same time, it should not increase the router pipeline latency. As ViChaR's [6] table based approach had solved several issues pertaining to latency and scalability, we have adopted a similar idea but limit the number of VCs.

Figure 3(c) explains the dynamically allocated router buffer proposed in this paper. We adopt the unified buffer structure and augment the architecture with a 'Unified VC State Table' (UVST). In this case, there are v VCs/port, z buffer slots/port and c channel buffers, with r approximately z/v. In the UVST, all elements are centrally located for every VC. Given the resource-constrained environment for NoCs, the size of this table is minimal and does not grow with the

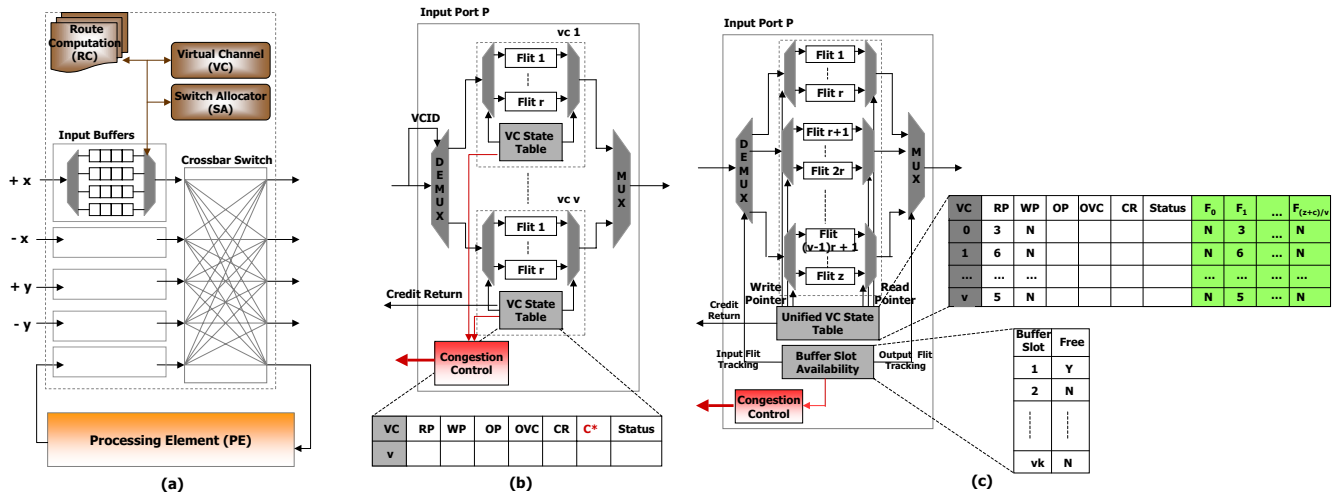


Figure 3: (a) A generic 5×5 NoC router architecture (b) Static buffer allocation with congestion control (c) Dynamic buffer allocation with congestion control.

number of VCs. The maximum size of the UVST is $O(v)$ as compared to that in ViChaR which is $O(vr)$. When a new flit arrives, its VCID cannot be used to switch as all buffer slots are unified. For that purpose, we use the 'Buffer Slot Availability' (BSA) tracking system to allocate/deallocate arriving/departing flits with buffer slots. Therefore, the DEMUX switches to the buffer slot provided by the BSA at the input flit tracking. BSA keeps track of all buffer slots currently available and allocates the first buffer slot found to be free. If the buffer slot number points to NULL, then such a slot can be selected for the newly arriving flit. After allocating the buffer slot to the incoming flit, BSA then searches for the next free slot to be allocated. Similarly, for a departing flit, BSA will de-allocate the buffer slot using the output flit tracking and add the free slot to the list of free slots maintained in the table (shown in the inset of Figure 3(c)).

Once the flit is associated with the input flit tracking number identifying which flit buffer it is destined to, the flit arrives at the second DEMUX. Here, the WP logic writes the flit to the buffer slot allocated by the BSA. In the same cycle, UVST identifies the VCID of the newly arriving flit and accordingly updates the UVST. If the newly arriving flit is the header flit, then it undergoes the usual stages of RC, VA, SA, and ST. The arbitration logic is similar to the generic case as there is no increase in the number of VCs. The table contains buffer slots F_0, F_1, ... $F_{(z+c)/v}$ in addition to the regular fields RP, WP, OP, OVC, CR and Status fields. The total number of credits is limited to $(z + c)/v$ per VC slot. The buffer slots are used to identify the location of the flit assigned to the particular VC. The number of buffer slots available depends on the maximum number of credits available for a particular VC. For fairness purposes, the number of credits is equally divided between all the different VCs. The responsibility for congestion detection rests with the BSA. When BSA does not find a non-null pointer in its base table, it triggers the congestion signal and when a free buffer slot is created by a departing flit, the BSA releases the congestion signal.

Dynamic buffer management is not as rigid as the static allocation scheme and eliminates the HoL blocking to some extent. If the number of VCs increased/decreased dynamically as in ViChaR, then the HoL blocking can be completely eliminated. Although our design can be extended to incorporate dynamic VC allocation as in ViChaR, this has not been explored in this paper. Our objective has been to reduce the HoL blocking existing due to the channel buffers. The channel buffers can be viewed as serial FIFO buffers as opposed to the parallel FIFO buffers used within the routers. Therefore, eliminating the HoL blocking is critical in our new design. Static allocation of buffer slots simplifies the overall design as it requires minimum extension over a generic NoC router architecture. Dynamic allocation of buffer slots along with the table based design significantly reduces the HoL blocking. This achieves much higher throughput while saving chip area and reducing the power consumption.

5. PERFORMANCE EVALUATION

In this section, we evaluate the router buffers and the proposed adaptive channel buffers in terms of power dissipation, area overhead and overall network performance, using 8×8 mesh and folded torus topologies. The design is implemented in 90 nm CMOS technology with a supply voltage of 1 V and an operating frequency of 500 MHz. The notation followed for the different cases is of $vn_V - rn_R - cn_C$, where n_V is the number of VCs per input port, n_R is the number of 128-bit router buffers per VC and n_C is the number of channel buffers. For example, the baseline is denoted as $v4 - r4 - c0$, implying 4 VCs/input port, 4 router buffers/VC and 0 channel buffers. The baseline design has 8 conventional repeaters along each wire of the 128-bit wide links.

5.1 Inter-Router Links

5.1.1 Link Power Estimation

The power per segment of a repeater-inserted link [3, 14] is given by

$$P_{segment} = P_{dynamic} + P_{leakage} + P_{short-ckt} \quad (1)$$

where $P_{dynamic}$ is the switching power, $P_{leakage}$ is the leakage power and $P_{short-ckt}$ is the short-circuit power. When a conventional repeater is replaced by a channel buffer (three-state repeater), there is an additional capacitance C_{buf}, due to the added transistors, as shown in Figures 1(a) and 1(b). This capacitance is estimated as

$$C_{buf} = (1/2 \times W \times L \times C_{ox}) + (W \times L_{ov} \times C_{ox}) \quad (2)$$

where W and L are the width and length of the minimum sized inverter respectively. C_{ox} is the oxide capacitance and L_{ov} is the gate-drain/source overlap length. The components of the total power for the channel buffer inserted line are calculated as

$$\widetilde{P}_{dynamic} = \alpha \times [k(C_o + C_p + C_{buf}) + \ell C_w] \times V_{DD}^2 \times freq \quad (3)$$

$$\widetilde{P}_{leakage} = 2 \times [1/2 \times V_{DD} \times (I_{off}(W_N + W_P)k)] \quad (4)$$

$$\widetilde{P}_{short-ckt} = \alpha \times (\ln 3 \times \widetilde{\tau}) \times W_N \times k \times V_{DD} \times I_{sc} \times freq \quad (5)$$

$$\widetilde{\tau} = R_s(C_o + C_p + C_{buf}) + (R_s/k)C_w\ell + R_w k C_o \ell + 0.5 R_w C_w \ell^2 \quad (6)$$

where $\widetilde{P}_{dynamic}$ is the dynamic power, $\widetilde{P}_{leakage}$ is the leakage power, $\widetilde{P}_{short-ckt}$ is the short-circuit power of the channel buffer inserted link, α is the activity factor, k is the repeater sizing, ℓ is the repeater spacing, W_N (W_P) is the width of the NMOS (PMOS) transistor in the repeater, C_o is the device diffusion capacitance, C_p is the device gate capacitance, C_w is wire capacitance, I_{off} is the subthreshold leakage current, I_{sc} is the device short-circuit current, V_{DD} is the supply voltage, $freq$ is the operating frequency, and $\widetilde{\tau}$ is the time-constant of the short-circuit current pulse in the channel buffer.

During congestion, the control block is switched 'ON' and power is dissipated in the inverters and the switched capacitor within the block.

$$P_{control-blk} = P_{sw-cap} + P_{inv} \quad (7)$$

where P_{sw-cap} is the power due to the switched capacitor and P_{inv} is the power due to the inverters. The block supplying the clock signals to the control blocks consists of inverters and NAND gates. The power consumed by this block, P_{clk} is the sum of the dynamic power, the leakage power and the short-circuit power of the individual gates. In the absence of congestion, the channel buffers function like repeaters and the power dissipated per segment of the link is $\widetilde{P}_{segment_{repeater}}$, and is given by

$$\widetilde{P}_{segment_{repeater}} = \widetilde{P}_{dynamic} + \widetilde{P}_{leakage} + \widetilde{P}_{short-ckt} \quad (8)$$

During congestion, the channel buffers store the data in position and the power dissipated is $\widetilde{P}_{segment_{chl-buffer}}$, and is given by

$$\widetilde{P}_{segment_{chl-buffer}} = \widetilde{P}_{leakage} + P_{control-block} + P_{clk} \quad (9)$$

In calculating the power values, power-optimal repeater insertion methodology described in [14] has been used. The inter-router links are assumed to be 2 mm long for the mesh network. In case of the folded torus network, the average channel length doubles [4] and hence the inter-router links are 4 mm long.

5.1.2 Link Area Estimation

The area consumed by the wires is determined as

$$Area_{wires} = N_W \times p_w \times s_w \quad (10)$$

where N_W is the the bit-width of the link, p_w is the wire pitch and s_w is the wire spacing at the given technology. The repeater area is calculated as

$$Area_{repeaters} = k \times Area_{min} \times N_R \times N_W \quad (11)$$

where $Area_{min}$ is the area of a minimum sized inverter at the given technology and N_R is the number of repeaters along the wire. When the conventional repeaters are replaced by the channel buffers, the area doubles due to the additional transistors in the channel buffers. The area overhead due to the control block for each channel buffer stage is the sum of the individual transistor areas in the block calculated as

$$Area_{control-block} = \sum (s \times Area_{min}) \quad (12)$$

where s is the transistor sizing in the control block. This area is negligible compared to the overall channel buffer area.

5.2 Router

5.2.1 Router Power Estimation

The dynamic power consumed by the buffer [18] (SRAM cell-array) is the power expended in writing a flit into the buffer and the power consumed to read out the flit from the buffer. The leakage power consumption of the buffer [19] is determined by the total leakage current in the buffer. Reducing the number of router buffers reduces the component capacitances within the buffer. For instance, the write-bitline capacitance is given by

$$C_{bw} = (B \times C_d(T_p)) + C_a(T_{bd}) + (C_w \times L_{bl}) \quad (13)$$

where B is the buffer size in flits, $C_d(T_p)$ is the drain-end capacitance of the write pass-transistor, $C_a(T_{bd})$ is the total capacitance of the bitline driver, C_w is the metal wire capacitance per unit length and L_{bl} is the bitline length. Reducing the number of router buffers reduces the bitline length and hence the load-dependent capacitance of the bitline driver. The crossbar and arbiter models [18] do not change across the different cases considered, since the flit width and the number of I/O ports in the router remain constant across all the cases.

5.2.2 Router Area Estimation

At the 90 nm technology considered, an SRAM cell has an estimated area of 1.0092 μm^2 [20]. Results from Intel's 90 nm technology also indicate an area of 1 μm^2 for the SRAM cell. The area of the crossbar is estimated by the number of input/output signals that it should accommodate [4]. Area of the arbiter is the area occupied by the two levels of NOR gates in the arbiter.

5.3 Comparison of the different cases

Table 1 shows a comparison of the power estimations for various router and channel buffer configurations. Change in power in each case is expressed as a percentage increase (+) or a percentage decrease (−) with respect to the baseline ($v4-r4-c0$). A maximum power savings of 37.02% and area savings of 50% is achieved for the third case that uses only half the number of router buffers compared to the baseline.

Table 1: Power Estimation for Various Channel and Router Buffer Configurations in a 8 × 8 mesh and a 8 × 8 folded torus interconnection network (Crossbar Power = 2.16 mW). Power values are for one flit traversal.

$vn_V - rn_R - cn_C$	Buffer Power (mW)	% Change	Mesh Link + Control Power (mW)	% Change	Folded torus Link + Control Power (mW)	% Change
v4-r4-c0	2.020	–	2.032+0	–	4.068	–
v4-r3-c4	1.646	-18.51	2.164 + 0.0122	+7.0	4.195 + 0.0122	+3.4
v4-r2-c8	1.272	-37.02	2.296 + 0.0205	+13.9	4.327 + 0.0205	+6.8
v3-r4-c4	1.646	-18.51	2.164 + 0.0122	+7.0	4.195 + 0.0122	+3.4
v3-r3-c7	1.365	-32.41	2.263 + 0.0184	+12.2	4.294 + 0.0184	+6.0
v5-r2-c6	1.459	-27.76	2.230 + 0.0164	+10.5	4.261 + 0.0164	+5.1
v5-r3-c1	1.926	-4.65	2.065 + 0.0059	+1.8	4.096 + 0.0059	+0.8

5.4 Simulation Methodology

A cycle-accurate on-chip network simulator was used to conduct a detailed evaluation of the proposed adaptive channel and router buffer design in both a 8 × 8 mesh and a 8 × 8 folded torus networks. The test configurations are represented in the results as $vn_V - rn_R - cn_C$, where n_V is the number of VCs per input port, n_R is the number of router flit-buffers per VC and n_C is the number of channel buffers. For simplicity, they will be referred to as $n_V n_R n_C$ in the following discussion. The test configurations evaluated were $n_V n_R n_C = 440$ (baseline), 434, 428, 344 and 531. We tested our hypothesis of using static and dynamic buffer allocation schemes on several traffic patterns such as: (1) Uniform Random (UN), where each node randomly selects its destinations with equal probability and (2) Permutation Patterns, where each node selects a fixed destination based on the permutations. We evaluated the performance on the following permutation patterns [13]: Bit-Reversal (BR), Butterfly (BU), Matrix Transpose (MT), Complement (CO), Perfect Shuffle (PS), Neighbor (NE) and Tornado (TO).

5.5 Simulation Results and Discussion

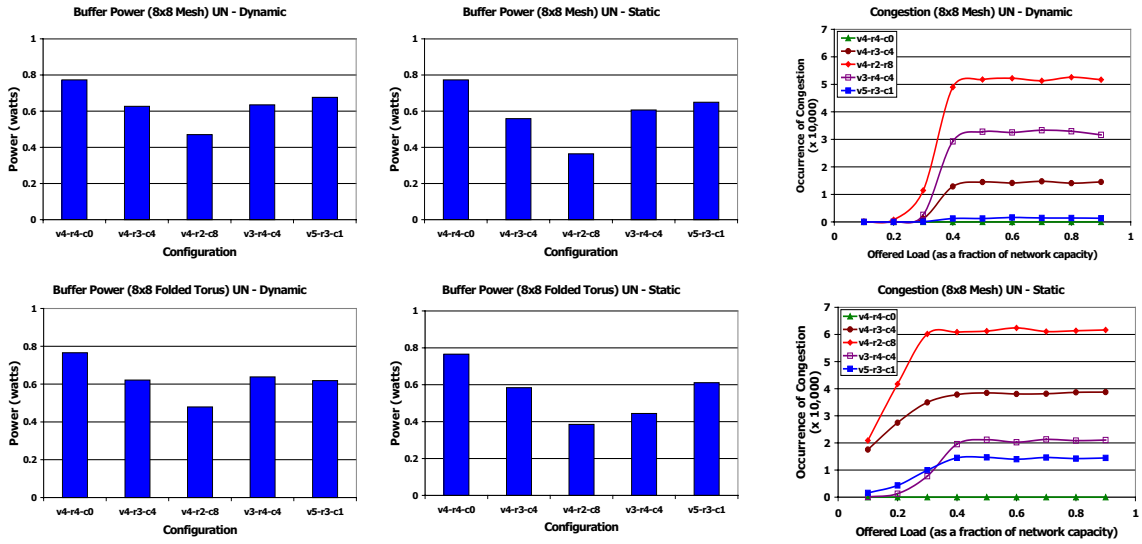
The following discussion presents the simulation results for the proposed adaptive channel and router buffer design in terms of the input buffer power consumed, the throughput, average latency and the occurrence of congestion in the network.

Input Buffer Power: Figure 4(a) shows the total power dissipated in the input buffers with static and dynamic buffer allocation for uniform (UN) traffic in the 8 × 8 mesh and the 8 × 8 folded torus networks for a network load of 0.5. For the mesh topology the power savings for the 428 configuration (reducing the buffer depth from 4 to 2) is nearly 40% under dynamic buffer allocation and about 52.5% under static buffer allocation. Similar results are observed for the folded torus topology with the 428 configuration achieving a power savings of nearly 37% for the dynamic case and 50% for the static case. Therefore in both static and dynamic cases, significant power savings is obtained by reducing the buffer size.

Congestion Variation: Figure 4(b) shows the occurrence of congestion over the total length of the simulation in the 8 × 8 mesh network for static and dynamic buffer allocations under uniform traffic. Congestion in the network indicates that the channel buffers are enabled to hold the data along the links. The baseline configuration (440) does not make use of channel buffers and therefore the occurrence of congestion is shown to be zero for the baseline in both the cases. The 428 configuration shows the highest occurrence of congestion due to the reduction of the router buffer size in half compared to the baseline. Static buffer allocation shows a higher occurrence of congestion than the dynamic case by about 18%, for the 428 configuration. Although the 428 configuration shows the maximum congestion in the network, the corresponding network performance does not drop significantly as the adaptive channel buffers hold the data along the links and prevent loss of data due to congestion.

Throughput, Latency and Power: Figure 5 shows the throughput, average latency and overall network power for uniform (UN) traffic using static and dynamic buffer allocation for varying network load, for the 8 × 8 mesh and the 8 × 8 folded torus networks. For dynamic buffer allocation in the mesh topology, the throughput shows almost similar performance for 440, 344 and 434 under uniform traffic. The 428 shows only about 3% drop in performance. This result is significant as we can save almost 50% of the buffer size and yet achieve similar performance as the baseline configuration by dynamically allocating the buffer resources to flits. At high network loads, the congestion signal prevents the data movement into the router buffer. As we have additional buffers in the channel, the flow of data flits is not hampered even though we have fewer buffers in the routers. For the static buffer allocation in the mesh network, the performance degradation compared to the baseline increases as the depth of the buffer is reduced, due to head-of-line (HoL) blocking. The poorly performing configuration is the 428, which shows about 20% reduction in throughput in both the mesh and the folded torus networks. The average network latency shown in Figure 5 reflects the HoL effects on various configurations in the mesh network. The total power consumed in the mesh network is shown in Figure 5 for a network load of 0.5, under uniform traffic. The 428 configuration shows a reduction of 20% in the overall network power under dynamic buffer allocation and nearly 27% under static buffer allocation.

From Figure 5, for the folded torus network using dynamic buffer allocation, all the configurations except 531 show similar performance for uniform traffic. This is significant since a reduction in the buffer depth or a decrease in the number of VCs has not degraded the performance compared to the baseline. For the static case the throughput drops by about 16% for the 428 configuration. Figure 5 also shows the total power consumed in the folded torus network for a network workload of 0.5 under uniform traffic. In case of dynamic buffer allocation the 428 configuration shows a decrease of about 27% in the total power. All the configurations achieve



(a) (b)

Figure 4: (a)Total power dissipated in the input buffers with static and dynamic buffer allocation for uniform (UN) traffic in 8 × 8 mesh and folded torus networks for a network load of 0.5. (b) Congestion under dynamic and static buffer allocation for uniform (UN) traffic in 8 × 8 mesh, over the total length of the simulation.

a reduction in power compared to the baseline, as seen in the case of the mesh network.

Throughput and Power for All Traffic Patterns: Figure 6 shows the power consumed at the input buffers and the throughput achieved at a network load of 0.5 for the 8 × 8 mesh network, with static and dynamic buffer allocation under all the traffic patterns considered, for 3 configurations, namely 440, 434 and 428. From Figure 6(a), it is can be observed that irrespective of whether the buffer allocation is static or dynamic, power savings is obtained for all traffic patterns. The power savings seen with static allocation is slightly more than the power savings observed with dynamic allocation. For the Complement traffic pattern, static buffer allocation provides 45% savings in input buffer power for the 428 configuration, while dynamic buffer allocation shows a savings of about 37.5%. Figure 6(b) shows no significant decrease in throughput for the dynamic case for all traffic patterns. Dynamic buffer allocation provides the flexibility for the flits to be allocated to any available buffer slot and is not as restrictive as the static allocation. For the Complement traffic, the dynamic buffer allocation provides similar performance for the 428 configuration as the baseline, where as with static approach, the loss in throughput is about 17%.

6. CONCLUSION

As recent research has shown that the design of the buffers in the router influences the energy consumption, area overhead and overall performance of the on-chip network, our design attempts at reducing the size of the buffers within the routers. As this impacts performance, we provide additional adaptive channel buffers which can be used when required. Simulation results show that by reducing the router buffer size in half, the adaptive channel buffer design leads to a 40-52% reduction in buffer power alone and more than 17-20% savings in the overall router power, with a significant 50% reduction in router area. There is only a marginal 1-5% drop in network performance for the design under dynamic buffer allocation and a 10-20% drop in performance under static buffer allocation. This paper shows that eliminating some of the buffers in the router and using adaptive communication channel buffers saves an appreciable amount of power and area, without significant degradation in the throughput or latency.

Acknowledgement: This research was partially supported by NSF grant ECCS-0725765. We thank Dr. Dongsheng (Brian) Ma and Minkyu Song for their assistance in analyzing the channel buffer implementation.

7. REFERENCES

[1] L. Benini and G. D. Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*, vol. 35, pp. 70–78, January, 2002.

[2] W. J. Dally and B. Towles, "Route packets, not wires," in *Proceedings of the Design Automation Conference (DAC)*, Las Vegas, NV, USA, pp. 684–689, June 18-22, 2001.

[3] R. Ho, K. W. Mai, and M. A. Horowitz, "The future of wires," *Proceedings of the IEEE*, vol. 89, pp. 490–504, April, 2001.

[4] J. Balfour and W. J. Dally, "Design tradeoffs for tiled CMP on-chip networks," in *Proceedings of the 20th ACM International Conference on Supercomputing (ICS)*, Cairns, Australia, pp. 187–198, June 28-30, 2006.

[5] J. Hu and R. Marculescu, "Application-specific buffer space allocation for network-on-chip router design," in *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, San Jose, CA, USA, pp. 354–361, November 7-11, 2004.

[6] C. A. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "ViChaR: A dynamic virtual channel regulator for network-on-chip routers," in *Proceedings of the 39th Annual International Symposium on Microarchitecture (MICRO)*, Orlando, FL, USA, pp. 333–344, December 9-13, 2006.

[7] H. S. Wang, L. S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in

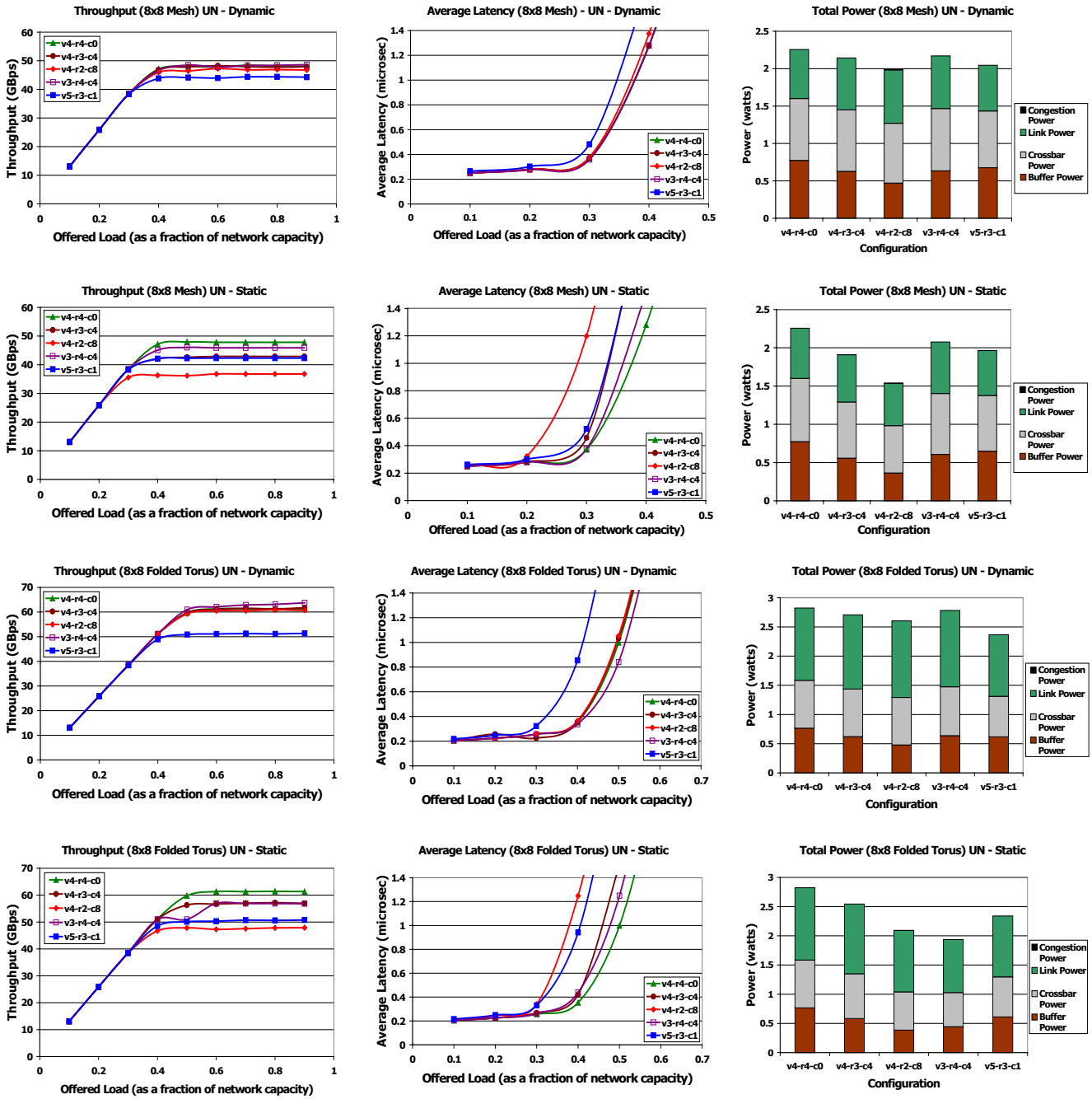
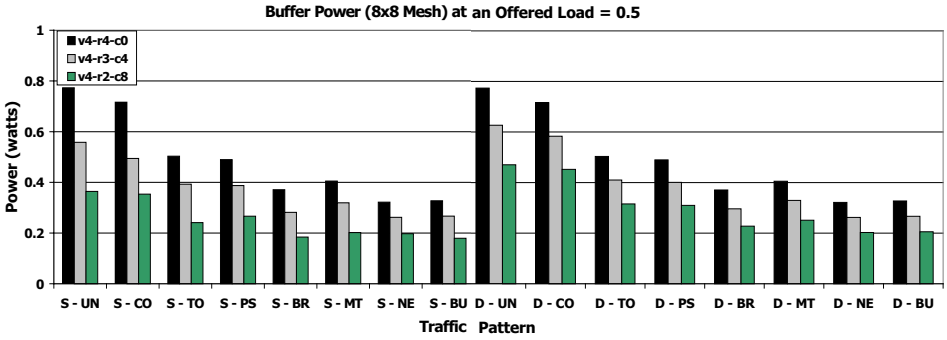
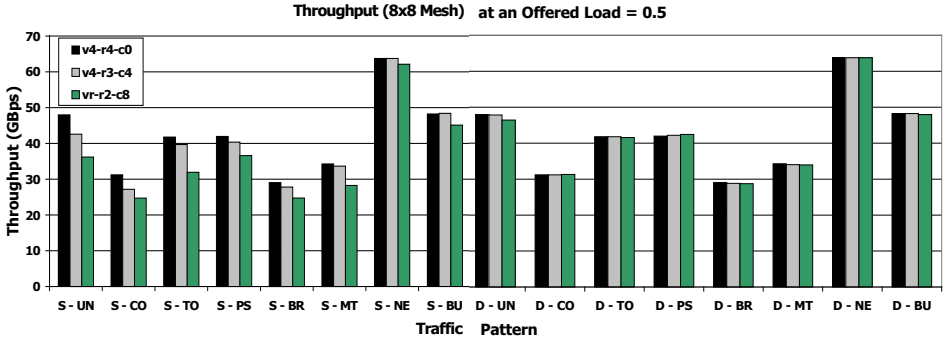


Figure 5: Throughput, latency and overall network power for uniform (UN) traffic using static and dynamic buffer allocation for varying network load for 8 × 8 mesh and 8 × 8 folded torus networks. The configurations tested were $vn_V - rn_R - cn_C$, where n_V is the number of VCs per input port, n_R is the number of router flit-buffers per VC and n_C is the number of channel buffers.



(a)



(b)

Figure 6: (a) Buffer power consumed and (b) Throughput at 0.5 network load with static (S) and dynamic (D) buffer allocation in the 8 × 8 mesh for all traffic patterns: Uniform (UN), Complement (CO), Tornado (TO), Perfect Shuffle (PS), Bit-Reversal (BR), Matrix Transpose (MT), Neighbor (NE) and Butterfly (BU).

Proceedings of the 36th Annual ACM/IEEE International Symposium on Microarchitecture (MICRO), Washington DC, USA, pp. 105–116, December 3-5, 2003.

[8] S. Heo and K. Asanovic, "Replacing global wires with an on-chip network: A power analysis," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED)*, San Diego, CA, USA, pp. 369–374, August 8-10 2005.

[9] "2006 workshop on on- and off-chip interconnection networks for multicore systems (http://www.ece.ucdavis.edu/∼ocin06/program.html)," December 6-7, 2006.

[10] J. Kim, C. A. Nicopoulos, D. Park, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "A gracefully degrading and energy-efficient modular router architecture for on-chip networks," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*, Boston, MA, USA, pp. 4–15, June 17-21, 2006.

[11] R. Mullins, A. West, and S. Moore, "Low-latency virtual channel routers for on-chip networks," in *Proceedings of International Symposium on Computer Architecture (ISCA)*, Munchen, Germany, pp. 188–197, June 19-23, 2004.

[12] L. S. Peh and W.J. Dally, "A delay model and speculative architecture for pipelined routers," in *Proceedings of the 7th International Symposium on High-Performance Computer Architecture (HPCA)*, Nuevo Leone, Mexico, pp. 255–266, January, 2001.

[13] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, San Fransisco, USA, 2004.

[14] K. Banerjee and A. Mehrotra, "A power-optimal repeater insertion methodology for global interconnects in nanometer designs," *IEEE Transactions on Electron Devices*, vol. 49, no. 11, pp. 2001–2007, November, 2002.

[15] M. Mizuno, W. J. Dally, and H. Onishi, "Elastic interconnects: Repeater-inserted long wiring capable of compressing and decompressing data," in *Proceedings of the IEEE International Solid-State Circuits Conference*, San Fransisco, CA, USA, pp. 346–347, February 5-7, 2001.

[16] N. Ni, M. Pirvu, and L. Bhuyan, "Circular buffered switch design with wormhole routing and virtual channels," in *Proceedings of the International Conference on Computer Design (ICCD)*, Austin, TX, USA, pp. 466–473, October, 1998.

[17] Y. Tamir and G. L. Frazier, "High-performance multiqueue buffers for VLSI communication switches," in *Proceedings of the 15th Annual International Symposium on Computer Architecture (ISCA)*, Honolulu, Hawaii, USA, pp. 343–354, May-June, 1988.

[18] H. S. Wang, X. Zhu, L. S. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection networks," in *Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture (MICRO)*, Istanbul, Turkey, pp. 294–305, November 18-22, 2002.

[19] X. Chen and L. S. Peh, "Leakage power modeling and optimization in interconnection networks," in *Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, Seoul, Korea, pp. 90–95, August 25-27, 2003.

[20] A. Y. Zeng, K. Rose, and R. J. Gutmann, "Cache array architecture optimization at deep submicron technologies," in *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors*, San Jose, CA, USA, pp. 320–325, October 11-13, 2004.