Adaptive Bubble Router: a Design to Improve Performance in Torus Networks*

V. Puente, R. Beivide, J.A. Gregorio, J.M. Prellezo, J. Duato*, and C. Izu**

Universidad de Cantabria
39005 Santander, Spain
{vpuente,mon,jagm,prellezo}@atc.unican.es

* Universidad P. de Valencia
46071 Valencia, Spain
jduato@gap.upv.es

** University of Adelaide
SA 5005 Australia
cruz@cs.adelaide.edu.au

Abstract

A router design for torus networks that significantly reduces message latency over traditional wormhole routers is presented in this paper. This new router implements virtual cut-through switching and fully-adaptive minimal routing. Packet deadlock is avoided by providing escape ways governed by Bubble flow control, a mechanism that guarantees enough free buffer space in the network to allow continuous packet movement.

Both deterministic and adaptive Bubble routers have been designed in VLSI using VHDL synthesis tools. Adopting a fair quantitative comparison, we demonstrate that Bubble routers exhibit a reduction in base latency values over 40% with respect to the corresponding wormhole routers, without any penalty in network throughput. With much lower VLSI costs than adaptive wormhole routers, the adaptive Bubble router is even faster than deterministic wormhole routers based on virtual channels.

Keywords: Interconnection subsystem, parallel computers, hardware routers, performance evaluation, VLSI design, simulation.

1 Introduction

Multiprocessor performance has considerably increased during the last decade. On the one hand, distributed shared-memory multiprocessors (DSMs) are becoming widespread (SGI Origin 2000 [16], Cray T3E [20]). On the other hand, nowadays message-passing multicomputers constitute the frontier of computing power (ASCI Project [15]). As processor computing power increases, communication performance should increase accordingly in order to adequately balance the system.

Interconnection networks have also significantly evolved, reducing message latency and increasing throughput. Most of these improvements come from architectural advances. A significant architectural advance has been the use of pipelined switching techniques. In particular, wormhole switching considerably reduces message latency by routing a message as soon as its header arrives at a router [7]. Data follow immediately after the header. This strategy splits messages into small units of information, or flits [6], performing flow control at the flit level. As a consequence of this unique feature, flit buffers can be very small, leading to compact and fast routers [4]. These benefits led to the implementation of wormhole switching in most commercial routers [2, 13, 19].

Moreover, this pipelined message transmission makes latency less sensitive to the distance in the network provided that messages are long enough, facilitating the search for optimal topologies. Several researchers recommended the use of low-dimensional direct networks in the k-ary n-cube class [1, 8]. As a result, the use of bidimensional or three-dimensional meshes and tori or limited-degree hypercubes is common in multicomputers and DSMs.

However, wormhole switching has also some disadvantages. A main one is that messages block in place when the link requested by the header is busy. So, data flits span over multiple routers, leading to significant link contention. Contention can be reduced by multiplexing physical bandwidth among several messages. This can be achieved by using separate buffers, or virtual channels [9], associated with each physical link. Virtual channels have also been proposed to avoid message deadlock [7]. Another way to reduce the negative effects of link contention consists of using adaptive routing, allowing packets to follow alternative paths. Fully adaptive routing also requires virtual channels to avoid deadlock in k-ary n-cube wormhole networks [10]. Finally, link contention can also be mitigated implementing buffers large enough to store full messages. For example, the Cray T3E router uses these three techniques: the virtual channel buffers have capacity for 12 and 22 flits, thus being able to store one and two messages, respectively. Moreover, adaptive virtual channels are added to the oblivious ones, only accepting a new message if there are enough empty flit buffers to store the whole message [19].

Other routers such as the Intel Cavallino [2], the SGI SPIDER [13] also employ a few virtual channels per phys-

*This work is supported in part by TIC98-1162-C02-01

ical link with the latter implementing fully adaptive routing as well. Both features improve throughput significantly [9, 10], and may reduce the execution time for bandwidth-limited parallel applications. However, virtual channels and adaptive routing have been shown to increment router delay [4], thus increasing the execution time of latency-sensitive parallel applications. For those applications, it has been suggested that routers should implement neither virtual channels nor adaptive routing [23]. Therefore, including virtual channels and adaptive routing in a wormhole router is a design tradeoff.

In this paper a new virtual cut-through (VCT) router is proposed. We show that router delay can be reduced with respect to a wormhole (WH) router while keeping the same functionality. We have considered a context similar to the one for the Cray T3E router, i.e., torus topology and short messages, developing complete router designs for WH and VCT switching. Both deterministic and fully adaptive minimal routing have been explored.

The contributions of this paper are: 1) a simple and efficient strategy to implement fully adaptive VCT routers, 2) a quantitative evaluation of deterministic and adaptive routers for wormhole and virtual cut-through flow control, showing that VCT router delay can be kept smaller than that of an oblivious WH router, and 3) an evaluation of 2-D and 3-D tori router networks under different traffic patterns, showing that VCT routers considerably outperform their WH counterparts.

The rest of this paper is organized as follows. Section 2 will present the framework and motivation for this research. An informal and intuitive description of our adaptive routing proposal will be presented in Section 3, followed by a description of the Bubble router architecture. Section 3 will also provide VLSI cost values for the proposed router as well as other router alternatives. A quantitative comparison of all routers will be presented in Section 4. Finally, Section 5 will summarize the findings of this work.

2 Motivation and Related Work

When comparing among routers he benefits of the faster ones are visible across all applications, whatever will be their network load, in a consistent fashion. In particular, for cc-NUMA machines, network latency at low load is critical because many applications use the network resources within this operational zone.

Base latency strongly depends on router complexity; any increment in router complexity implies either a decrement in the router clock frequency or an increment in the number of router pipeline stages. So, we should aim at designing simpler and faster routers with few or no virtual channels and simple routing algorithms. However, if we simplify router design, how do we reduce the contention that typically arises when executing very demanding parallel applications?

A way to reduce congestion without requiring virtual channels consists of using virtual cut-through switching [14]. This switching technique requires buffers with capacity for one or more packets, thus removing blocked packets from the network. However, buffer requirements prevented the widespread use of VCT in multiprocessor interconnection networks. As mentioned before, recent wormhole-based routers already incorporate large buffers to deal with contention. Thus, the main difference, between VCT and WH routers is the size of their flow control unit. VCT performs flow control at the packet level instead of at the smaller data units employed in WH routers. Moreover, DSMs only require the transmission of very short messages (10 flits at most in the Cray T3E [19]) so buffer requirements can be kept reasonably low. This scenario is also similar for cc-NUMA machines.

Our goal is to design a fast VCT router which will provide lower latency than current wormhole routers at any working load. A number of reasons support this purpose. To start with, VCT flow control is simpler than WH and therefore faster when implemented. Secondly, if we consider the traffic properties under VCT flow control, we could simplify deadlock avoidance as shown in [3], reducing the required number of virtual channels under deterministic routing from two down to one. We will take this approach, which increases router speed as well. To make our router competitive at medium and high loads, adaptivity should be achieved with minimal implementation cost. The simplest fully adaptive wormhole router [11] implies the addition of one or more virtual channels to an existing deadlock-free network. Similarly, we could add one queue (virtual channel) to the deadlock-free proposal cited above, thus obtaining adaptivity at the minimum possible cost.

With respect to other related work, most of the research effort in the last decade has focused on improving wormholed-based routers by means of different routing and deadlock avoidance/recovery strategies. Little has been done, though, for virtual cut-through networks. To the best of our knowledge, few proposals, apart from our own, have considered $VCT's$ impact on network behavior ([18] [5] [17] among others): the more close to our proposal are the Chaos router [18], which implements non-minimal adaptive routing, and the minimal adaptive routing proposed by Cypher and Gravano [5].

3 The Bubble Router

In this section we propose a minimal fully adaptive routing algorithm for k-ary n-cube networks using virtual cut-through switching. After an informal description of this algorithm, we describe the VLSI design of the Bubble router, discussing low level issues.

3.1 Adaptive Routing Algorithm for Torus Networks

We will present our adaptive routing algorithm informally. A formal description and subsequent proofs have been developed but they have been omitted in this paper for the sake of simplicity.

Consider a full-duplex torus network; each link can be viewed as two unidirectional links in opposite directions. Each unidirectional link has two FIFO queues associated to its input edge. These queues will be referred to as the *adaptive* and the *escape* queue, respectively. Bidirectional routers can be simply obtained by duplicating links and queues for the opposite directions as well as increasing the crossbar size. Routers for n-dimensional torus can also be easily obtained by extending the basic router resources, i.e., links with their associated queues and crossbar size. Each of the input queues is conceptually identical to a virtual channel in a wormhole-based router. As we assume VCT switching, each queue must be able to store at least one packet: thus, we measure the capacity of a queue in terms of packet units.

Our adaptive algorithm is based on combining two strategies: dimension-order routing (DOR) with Bubble flow control in the escape queues [3], and fully adaptive minimal routing in the adaptive queues [12]. Packets can move from an adaptive queue to an escape queue or viceversa, fulfilling the rules described below. If both queues are available, preference is given to the adaptive one. Packets using a escape queue can freely use, if available, an adaptive queue at the next router. The resulting routing algorithm is minimal fully adaptive and reduces hardware requirements with respect to previous proposals. As a result, router delay is considerably reduced, as it will be shown in the next subsection.

Packets travelling through escape queues use DOR paths and are regulated by Bubble flow control. Bubble flow control works as follows. Consider the set of unidirectional links along a given direction in a dimension of a n-dimensional torus network. Those links form a unidirectional ring as can be seen in Figure 1. Considering only the escape queues, packets (shaded units at each queue) are allowed to progress (shaded arrows) to another queue along the ring if there is an empty packet unit at that queue, that is, there is enough buffer space to store the whole packet. Note that this is the basic requirement of virtual cut-through switching. Thus, no additional constraints are imposed at this point. However, packet injection into the ring is only allowed if after injection there is at least one empty packet unit in the set of queues for the whole ring corresponding to the dimension (and direction) requested by the incoming packet. That is the key to avoid deadlock in any ring because the empty packet unit acts as a *bubble* which allows continuous packet movement along the ring. Although a number of solutions to provide a bubble exist, in our final implementation, before allowing a new packet injection in a ring at an arbitrary node, we check for the existence of two free packet units in the escape queue at that node corresponding to the requested ring (*Bubble Condition*).

Similarly, when a packet attempts to change to the next dimension, it will be granted access to the escape queue if there are at least two empty packet units in the requested ring. Thus, moving to another dimension is treated as a packet injection.

In short, Bubble flow control avoids deadlock inside rings. Additionally, it is well known that DOR avoids deadlock involving several dimensions because they are crossed in order and there are no cyclic dependencies between buffer resources [7]. Therefore, the whole set of network escape queues is deadlock-free.

Packets travelling through adaptive queues can use any minimal path and are regulated by virtual cut-through flow control. Packets can use adaptive queues without any constraint provided that there is enough space for one packet. This applies to any packet, independently of where it comes from. Bubble flow control will only be applied if the next selected queue is an escape one. When both the adaptive and escape queue of a profitable channel are available, priority is given to the adaptive one. This selection policy provides a high routing flexibility, allowing packets to follow any minimal path available in the network.

The adaptive routing algorithm described above is also deadlock-free. As indicated in [12], adding queues to a deadlock-free routing algorithm cannot induce any deadlock, regardless of how packets are routed in the additional queues. The only constraint is that the routing algorithm should not consider the queue currently storing the packet when computing the set of routing options. This constraint is met by our routing algorithm because a packet can be routed on both escape and adaptive queues at any router, regardless of the queue storing it. Taking into account that DOR with Bubble flow control is deadlock-free, it follows that the whole routing algorithm is also deadlock-free.

When all the adaptive queues are full, as in Figure 2.(a), the network behaves as if only escape queues existed, as shown in Figure 2.(b). From the point of view of the escape queues, this state is identical to that depicted in Figure 1. It is clear from Figure 2 that escape queues are used by the packets stored in adaptive queues, avoiding any deadlock situation in the network with only two queues per physical link.

A VCT network which applies only Bubble flow control may lead to starvation. This problem arises in deterministic routing because packets advancing along a given unidimensional ring have a higher priority than those that are trying to enter that ring. For example, in Figure 3, packets going from router A to router D have a higher priority than packets trying to be injected at router B. Thus, if traffic between A and D is not stopped, a packet can be indefinitely waiting to enter at B. However, starvation cannot occur for the adaptive Bubble algorithm because the access to the adaptive queues is not restricted. This means that

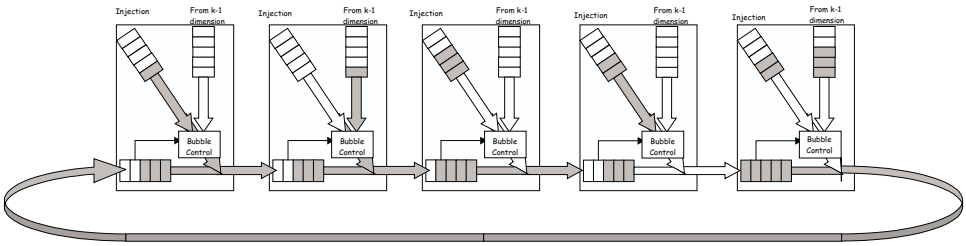

Figure 1. Deadlock avoidance in a ring by using Bubble flow control.

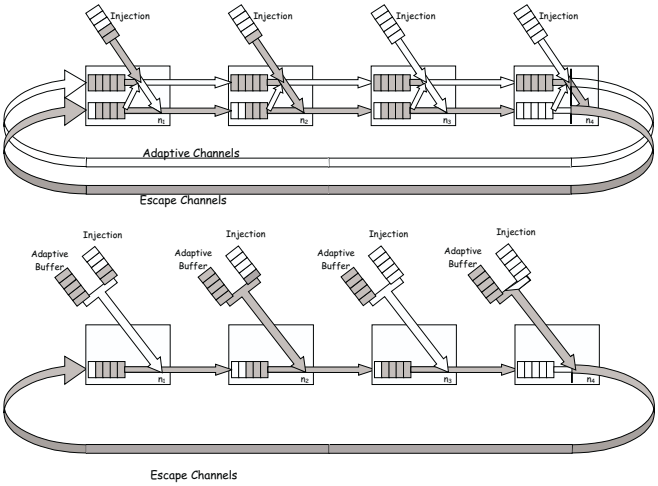

Figure 2. Representation of adaptive and escape queues when all adaptive ones are busy and equivalent network.

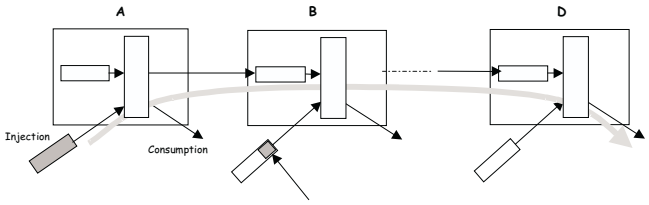

Figure 3. Example of starvation in deterministic Bubble flow control. It is avoided in our proposal.

packets in transit have the same probability to acquire the adaptive queue as incoming packets. In the previous example, the packet to be injected at B will succeed entering next node's adaptive queue. Even if that queue is temporarily full, a packet unit will eventually become free because the network is deadlock-free.

3.2 Bubble Router Design

In order to validate our proposal, a router for bidirectional n-dimensional torus networks has been designed. In particular, this router has been specified using the hardware description language VHDL. Internal router components are clocked synchronously but communications with neighbor routers are asynchronous in order to avoid clock skew problems.

This section describes the design of the Bubble router, highlighting the differences with respect to other design alternatives.

3.2.1 Basic Router Blocks

The main blocks of the proposed router architecture are shown in Figure 4. There is an escape queue and an adaptive queue for each physical link. They behave as described in the previous section. Both of them are implemented as FIFO queues with an additional *synchronization* module to support the asynchronous communication between routers.

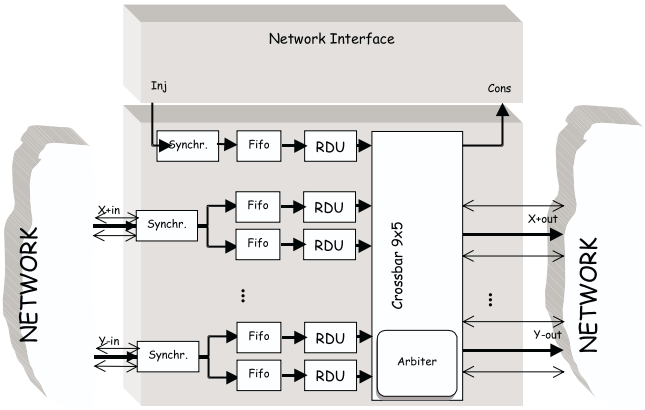

Figure 4. Router organization for a 2-D torus.

Input queues and output ports are connected through a crossbar. A round-robin policy has been chosen to select among packets contending for the same output link. Moreover, taking into account that link multiplexing is performed on a packet-by-packet basis, a virtual channel controller is

not required at the crossbar output. This reduces the latency of packets crossing the router. Besides, it is not necessary to send acknowledgment signals for every transmitted phit, and therefore, communication is less sensitive to wire delays.

Physical data links (and phits) in the current implementation are 9 bits wide, 8 bits for data and 1 bit for indicating the packet tail. Packets are 16 phits long, including the n-phit header which represents the destination address as a tuple of n offsets, i.e., the x and y offsets in a 2D torus.

The *routing decision unit* (RDU) associated to each input queue selects the output port taking into account the information in the packet header, the available local output ports and the status of the neighbor's queues. This status is known through external signals which are processed by the crossbar arbiter. If the selected output port and queue are available, the RDU updates the header phits so that the offset in the selected dimension is decremented, and sent first. This decrement operation is carried out simultaneously for x and y offsets, and in parallel with the crossbar arbitration. However, only one of the modified offsets will be forwarded as the first phit according to the selected output.

In order to keep the arbiter complexity low, potential output ports are requested in a cyclic sequential manner, one port per cycle until one of them is selected. Although this strategy may seem to decrease throughput, our experiments showed no noticeable degradation [22].

Queues are requested to the crossbar arbiter in the following order:

1. The adaptive queue of the neighbor along the incoming dimension, if the first offset is not zero.

2. The adaptive queue of the neighbor along the other dimension, if the second offset is not zero.

3. The escape queue of the neighbor along the first dimension, if the x offset is not zero or the escape queue of the neighbor along the second dimension if the x offset is zero. Before requesting an escape queue, the RDU verifies the Bubble condition. To do so, it checks that there are at least two empty packet units at the current router's escape queue. To guarantee the existence of one free packet unit it needs to ask for two; the latter may be acquired during that cycle by a transit packet.

Therefore, the routing decision in an empty network will take one cycle except for the $x - to - y$ turn and the delivery to the consumption channel. Both cases need two cycles as they examine both header phits. Finally, it is worth mentioning that the operations carried out by the routing decision unit depend on packet destination but they do not depend on the queue storing the packet. Thus, the RDU is identical for both adaptive and escape queues.

3.2.2 Time and Area of the Current Design

We have designed the entire Bubble router using the *Synopsys v1997.08* synthesis tool. Then, we mapped our design into 0.7 μm (two metal layers) technology from ATMEL/ES2 foundry under typical working conditions with standard cells from *Es2 Synopsys design kit V5.2*. Although the obtained results for area and time are estimated by *Synopsys* in a pessimistic way, they provide us with values very close to the physical domain.

Table 1 shows both delay and area for each one of the router blocks. As the design is pipelined, the block in the critical path with highest delay determines the maximum clock frequency of the device. In this particular case it is the crossbar, due to its arbitration complexity, which imposes a cycle time of 5.65 ns. For the queue blocks we have considered three possible sizes: 32, 64, and 128 phits, in order to see the impact that queue depth has on its management time. For this design, 128 phits is the optimal value because it is the largest queue whose access time is still lower than the selected cycle time. Nevertheless, as we will see below, we actually use queues 64 phits long.

Module	**Critical Path (ns)**	**Area(mm^2)**
Sync.	3.53	0.51($\times$5)
Queue (32 phits)	5.19	0.41
Queue (64 phits)	*5.22*	*0.59*($\times$9)
Queue (128 phits)	5.25	0.94
Routing Dec. Unit	5.64	0.80($\times$9)
Crossbar & Arbiter	5.65	3.70($\times$1)
Total (64 phits)	5.65	18.76

Table 1. Characteristics of the router modules under typical conditions.

These results lead to the number of clock cycles represented in Figure 5. Crossbar and FIFO stages consume one cycle each. One or two cycles are spent in the routing decision unit depending on the direction followed by the packet (turns and consumption are penalized with an additional cycle). Finally, because of its asynchronous behavior, the synchronization module spends a variable amount of time (1 cycle on average).

3.3 Design of Alternative Router Organizations

Once the new router has been designed, the next step is to assess its performance for specific networks. However, it is not enough to obtain absolute performance figures for our router. We also need to compare it with other proposals using the same design methodology.

To achieve this goal, three additional routers have been designed for the same topology. Their organizations represent standard solutions for routers to be included into toroidal networks.

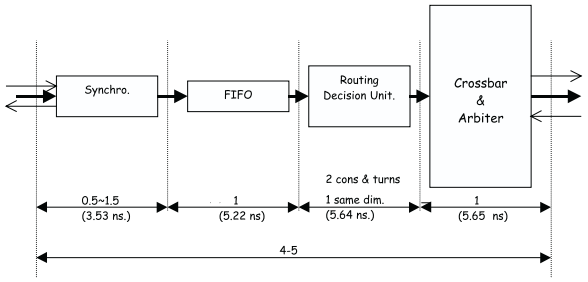

Figure 5. Delay in clock cycles for each module in the Bubble router pipeline.

A fully adaptive wormhole router that uses Duato's routing algorithm [11] for avoiding message deadlock has been designed . It requires at least three virtual channels per link. The crossbar size is 13×13 for a two-dimensional router. As in the router for the Cray T3E, wormhole switching is used but the maximum message length is limited. By doing so, the restriction on the use of adaptive channels can be eliminated, and therefore, the presence in the same queue of flits belonging to different messages is allowed. Although other alternatives for flow control have been tested, this is the one that offers the best results.

Additionally, in order to measure the cost of adaptivity, two routers using deterministic routing (DOR) have been designed. The first one, based on [6], uses wormhole switching and two virtual channels to avoid deadlock. We have not used more advanced algorithms for handling virtual channels, like the one proposed in [11], because it increases the router complexity and diminishes the main advantage of deterministic routers: low latency. The second deterministic router is based on the Bubble algorithm [3], with only one set of queues which follows the same flow control strategy as the escape queues described in Section 3.1.

The pipelined structures for wormhole routers are shown in Figure 6. The results for area and cycle time for each of these three designs, obtained from the hardware synthesis tools, are shown in Table 2. Note that the building blocks differ in each implementation as it was described in each original proposal but we apply the same VLSI design style. Additionally, due to the long arbitration delay in the adaptive wormhole router, the crossbar and arbiter stage has been split into two stages. To fairly compare among designs, we assume a total buffer capacity for the set of queues attached to each input channel of 128 flits for WH routers and 128 phits for VCT routers. It can be seen in Table 2 that this buffer capacity penalizes the clock cycle value for the Bubble DOR implementation. The small difference in time between the crossbar and arbiter units of both wormhole VC DOR and Bubble adaptive routers (Table 1)is due to the later having to control channel multiplexing.

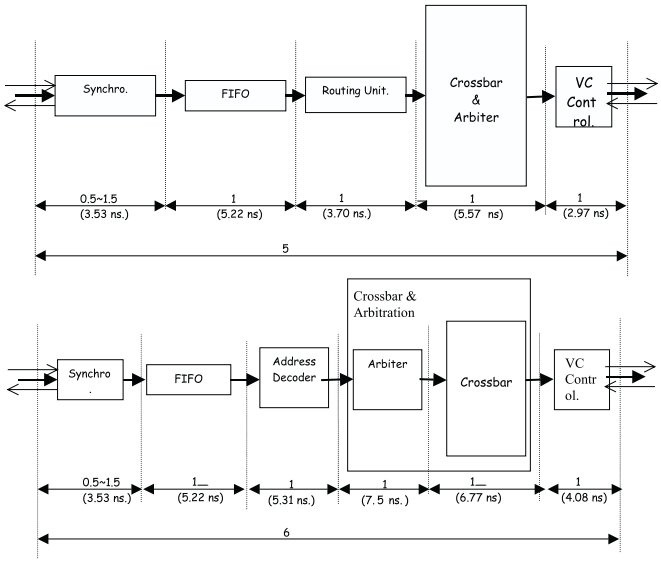

Figure 6. Pipeline structures for VC DOR and VC Adaptive routers.

4 Quantitative Router Evaluation

In this section, we compare the Bubble router performance with those of the alternative routers. First, we will compare router delay and its impact on network base latency based on the time analysis presented above. Then, we will analyze router behavior inside specific networks.

4.1 Evaluation of VLSI Router Cost

Table 3 summarizes, for all the routers, the required silicon area and time delays presented in the previous section. It is clear that methods based on Bubble flow control are, at least, a realistic alternative to the classical wormhole solutions based on virtual channels.

In our designs, the circuit's critical path is the arbiter plus crossbar unit except for the VC adaptive router in which the arbiter and the crossbar are decoupled, being the arbiter the stage that dictates the cycle time. This time is strongly related to the number of virtual channels that can request a given output port. Therefore, the cycle time for our *adaptive* Bubble router is similar to that of the *deterministic* wormhole router. Traditional wormhole routers require a virtual channel controller in order to provide link multiplexing at the flit-level. Bubble-based routers eliminate one pipeline stage per node because, as mentioned above, they do not require a virtual channel controller at each output port. This reduces silicon area for the adaptive Bubble router which uses only 56% of the required area for the adaptive wormhole counterpart. Besides, a shorter pipeline also reduces the hop time.

	Bubble DOR (*VCT*)		VC. DOR (*WH*)		VC. Adaptive (*WH*)	
Module	Time($ns.$)	Area(mm^2)	Time($ns.$)	Area(mm^2)	Time($ns.$)	Area(mm^2)
Sync.	3.53	0.51($\times$5)	3.53	0.51($\times$5)	3.53	0.51($\times$5)
Standard Queue	**5.25**	0.94($\times$4)	5.22	1.18($\times$4)	5.22	1.51($\times$4)
Injection Queue	**5.25**	0.94($\times$1)	5.25	0.94($\times$1)	5.25	0.94($\times$1)
Crossbar & Arbiter	4.31	2.15($\times$1)	**5.57**	6.30($\times$1)	6.77,**7.50**	7.86($\times$1)
Routing Unit	3.58	0.13($\times$5)	3.70	0.06($\times$9)	–	–
Address decoder	–	–	–	–	5.31	0.86($\times$13)
VC Controller	–	–	2.97	0.17($\times$ 4)	4.08	1.20($\times$4)
Total	5.25	10.05	5.57	15.73	7.5	33.37

Table 2. Characteristics of alternative router designs for 2-D tori.

Router	**Clock Cycles**	**Crossbar Size**	Queue **Sizes**	**Cycle time** ($ns.$)	**Area** (mm^2)	**Hop Time** ($ns.$)
Bubble DOR	4	5×5	128	5.25	10.05	21.0
VC DOR	5	9×9	$64 + 64$	5.57	15.73	27.85
VC Adap.	6	13×13	$64 + 32 + 32$	7.5	33.37	45
Bubble Adap	*4-5*	9×5	$64 + 64$	*5.65*	*18.76*	*25.43*

Table 3. Characteristics of router designs for 2-D tori.

Although a hardware design of a Bubble 3-D router has not yet been completed, preliminary results have been obtained by designing in VLSI the most critical parts of the routers. The additional costs are basically due to the increment in the number of inputs and outputs links. Thus, a 3-D router increases its pin-count by a factor of 7/5 with respect to a 2-D counterpart; obviously, this produces an increment in both silicon area and clock cycle.

Taking into account the bottlenecks found in 2-D routers, we encountered that the most critical component was the crossbar arbiter, and therefore, we paid more attention to the crossbar unit. In fact, we have developed VHDL instances of each one of the necessary crossbars for the four 3-D routers under analysis. Table 4 shows preliminary results.

Router	Clock cycles	Crossbar size	Crossbar critical path	Router cycle time
Bub. DOR	4	7×7	4.92	5.25
VC DOR	5	13×13	5.79	5.79
VC Adap	6	19×19	8.71	8.71
Bub. Adap	5	13×7	6.28	6.28

Table 4. Main characteristics of 3-D torus routers.

4.2 Network Analysis

Besides comparing stand-alone routers, we should compare their behavior when used to build specific networks. Although the physical implementation of the whole network would be the best way to assess its performance, accurate results can also be obtained by employing simulation tools.

Simulations have been performed at low level, using *Vhdlsim* and *Leapfrog*, VHDL simulators from *Synopsys* and *Cadence* respectively. In this way, every part of the router is accessible and can be monitored. However, this method has the drawback of requiring long simulation time. For instance, the average time for simulating 20000 clock cycles of an 8×8 2-D torus is about 10 hours running *Vhdlsim* and 2.5 hours for *Leapfrog* in the same conditions on an UltraSparc2 with 128 Mbytes.

With the aim of reducing the design cycle, an object oriented simulator has been written in C++. This simulator, called *SICOSYS (SImulator for COmmunication SYStem)* [21], has a remarkable feature: low-level characteristics of each router have been incorporated into the code, including the component delays obtained by using VLSI tools. This approach guarantees the delivery of accurate results which are very close to those obtained through VHDL simulation (discrepancies in latency less than 4%).

The results presented below have been obtained using *SICOSYS* and applying the component delays presented in section 3. The results correspond to 16-ary 2-cube networks under two different message destination patterns: random and specific permutations. In the first case, all the nodes have the same probability of becoming the destination for a given message. Two message length distributions were

considered for this pattern: only short messages (16 phits[1]) and a mix of short and long messages in order to simulate bimodal traffic (16 phits and 160 phits[2], respectively). The latter resembles the traffic generated in DSM multiprocessors in which short messages correspond to requests (or invalidation primitives for cc-NUMA), and long messages correspond to cache lines.

For the specific permutation experiments, three traffic patterns were taken into account: matrix transpose, bit reversal, and perfect-shuffle. All three cases are frequently used in numerical applications, such as in ADI methods to solve differential equation systems and in FFT algorithms, among others. In all the experiments, the traffic generation rate is uniform and randomly distributed over time.

4.2.1 Network Latency at Zero Load

Base latency depends on the router complexity. Therefore, adaptive routers tend to exhibit lower performance in low load situations. In our proposal this drawback has been minimized in two ways: limiting the number of queues per link to only two, and implementing a sequential arbiter to manage the requests for the router output links.

Table 5 shows the base latency for all the router designs and traffic patterns considered. The corresponding values have been obtained when the load applied to the network was around 0.05% of the bisection bandwidth.

Router	Rand	Bimo	Mtra	BitR	PerfS
Bub. DOR	276.5	358.2	284.2	282.2	279.7
VC DOR	338.6	424.6	348.5	346.2	342.8
VC Adap.	515.4	652.2	543.7	539.5	518.5
Bub. Adap	303.8	406.6	318.8	316.3	306.1

Table 5. Base latency (ns).

As we have previously mentioned, our main goal is to reduce the latency values in the most frequently used scenarios of network operation because many parallel applications for DSM machines are latency-limited, thus benefiting from reductions in message latency [23]. As expected, the deterministic Bubble router, which has the lower through delay, exhibits the lowest network base latency. Even when adaptivity is supported, the adaptive Bubble router exhibits latency values better than those obtained by deterministic wormhole routers. On the contrary, supporting fully adaptive routing in a wormhole-based router comes at a high cost: an increment of more than 50% in base latency.

4.2.2 Maximum Sustained Throughput

In spite of message latency being the main figure of merit in order to reduce the execution time of most parallel applications, another metric like the maximum throughput is frequently used to measure the performance of a specific interconnection network. Determining the saturation point of the network is nearly impossible in our context because this value strongly depends on the simulation conditions such as the random generation of values and the seeds election. Notwithstanding, the maximum value of the accepted traffic can give an idea of the network behavior in congestion conditions. These values expressed in structural terms (phits accepted per network cycle) and technological terms (phits accepted per nanosecond) can be seen in Table 6 for the different traffic patterns and routers. In all cases, the Bubble adaptive router achieves the largest throughput when measured in phits/ns.

4.2.3 Network Latency and Throughput at Variable Load

To complete this quantitative comparison, we need to consider network behavior under different message patterns and variable network loads. To justify the additional cost of adaptivity, we need to evaluate both deterministic and adaptive routers at medium to high workloads.

For space restriction only can show in Figure 7 the plots for the average message latency and throughput for 256-node 2-D torus versus applied load which is expressed only in phits per nanosecond for matrix transpose traffic. It should be noted that latency values include injection buffer delays. Injection delays must be taken into account in order to produce reliable results because they depend not only on the traffic load but also on the traffic pattern.

These results clearly show the superior behavior of both Bubble-based routers over wormhole-based counterparts in torus networks. The differences in latency values observed for zero load are maintained along the workable load range. For loads close to network saturation, all routers exhibit latency values which are even up to five times greater than the original base latency. This values can be prohibitively large for most parallel applications.

Obviously, for random traffic (see Table 6) the Bubble DOR router outperforms its adaptive counterpart but only by a small amount. It should be noted, though, that the adaptive Bubble router shows a behavior similar or even better than the VC DOR router, and clearly outperforms the VC adaptive one. However, a random destination traffic with a bimodal length distribution is more susceptible to congestion and benefits from adaptivity.

As it can be seen in Tables 5, the adaptive Bubble router reduces latency for medium to high loads. Furthermore, the adaptive Bubble router considerably improves network throughput when non-uniform traffic is being considered, consequently expanding the range of network load that can be efficiently handled. This is interesting for bandwidth-limited parallel applications which are expected to generate workloads close to the saturation region. The VC adaptive router achieves slightly higher throughputs in structural terms (see Table 6) for most permutation patterns but this

[1] In wormhole routers 1 phit = 1 flit

[2] in VCT routers the long messages are fragmented in 10 smalls packets

Traffic	Random		Bimodal		M. Transpose		P.Shuffle		Bit Reversal	
Router	Phits/cycle	Phits/ns	Phits/cycle	Phits/ns	Phits/cycle	Phits/ns	Phits/cycle	Phits/ns	Phits/cycle	Phits/ns
Bub. DOR	88.35	16.83	68.86	13.12	28.67	5.46	32.13	6.12	24.32	4.63
VC DOR	75.14	13.49	52.55	9.97	30.10	5.40	27.14	4.87	28.08	4.50
VC Adap	98.68	13.15	91.90	12.50	60.41	8.05	48.38	6.42	85.25	11.37
Bub. Adap	94.84	16.79	83.32	14.74	54.78	9.70	43.78	7.49	74.24	13.14

Table 6. Maximum achievable throughput for 2-D networks for the four routers under various traffic patterns.

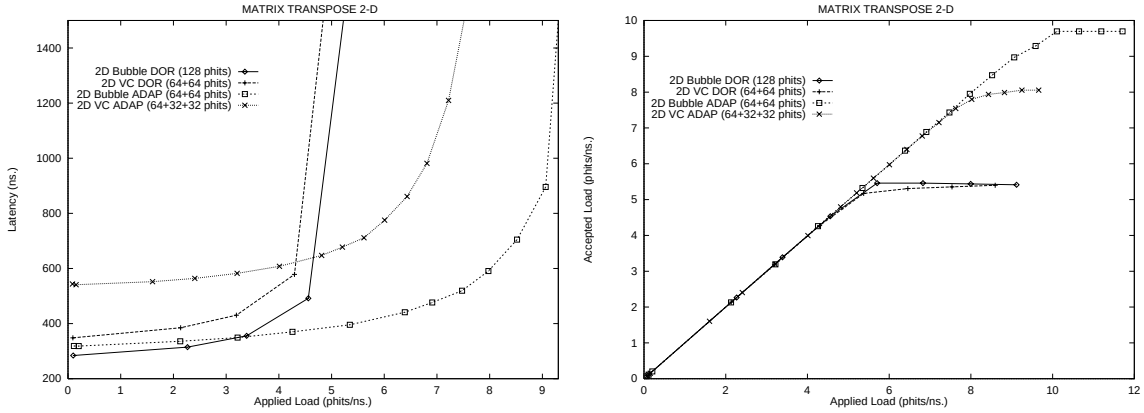

Figure 7. Latency and Throughput for a 256-node 2-D Torus under Matrix Transpose Traffic.

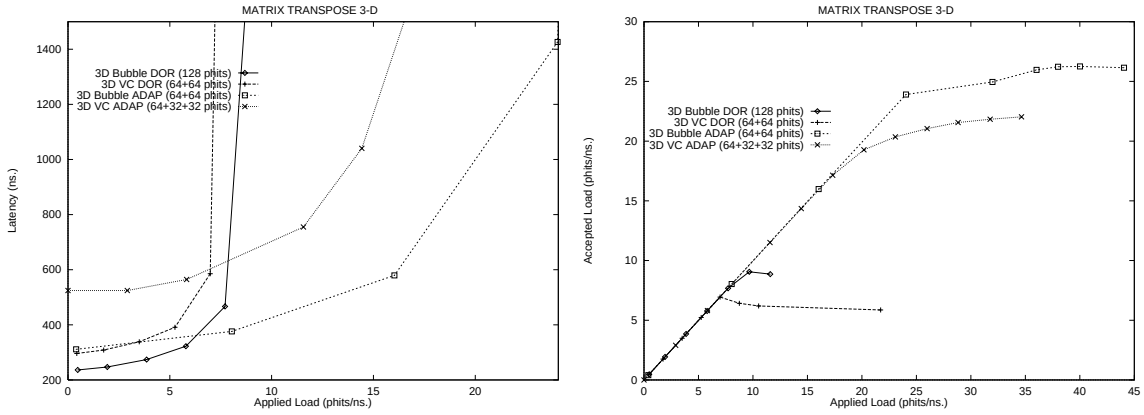

Figure 8. Latency and Throughput for a 512-node 3-D Torus under Matrix Transpose Traffic.

gain disappear if the cycle time is taken in account.

We have evaluated the four routers in a 3-D torus topology, which is the case for the Cray T3E router. As indicated above, these simulations use delay values from preliminary VLSI designs, and further optimization is in progress. Figure 8 shows the results on a 512-node 3-D torus with matrix transpose traffic. Although the adaptive Bubble router exhibits 10% higher base latency than the deterministic wormhole router, it is still a very competitive alternative, specially for non-uniform workloads for which it exhibits lower latency than the adaptive wormhole router (in the order of 40%) and higher throughput.

5 Conclusions

We have presented in this paper a simple mechanism to implement fully adaptive virtual cut-through routers, based on Bubble flow control. This flow control technique guarantees continuous message movement through any ring in the network by preserving at least one empty packet unit in the ring's queues. The queues that apply this deadlock avoidance strategy are used as escape ways for messages. Thus, DOR routing in the escape queues can be combined with additional queues using any other adaptive routing algorithm. Consequently, the Bubble router employs deadlock-free fully adaptive minimal routing for torus networks of any number of dimensions.

The obtained results demonstrate noticeable performance gains for the proposed router with respect to routers based on wormhole flow control and virtual channels. Our Bubble router design can be implemented with lower costs, in silicon area and delay, than those for wormhole routers based on virtual channels. The delay per hop in the Bubble routers is lower than those exhibited by both deterministic and adaptive wormhole routers. This translates into better network performance at most loads. Despite being a fully adaptive router, the latency exhibited by the adaptive Bubble router is similar to, or in most cases, lower than that of the deterministic wormhole router. Besides, it clearly outperforms the adaptive wormhole router. For instance, under random traffic we obtain gains in base latency of 41% and 37% for 16-ary 2-cube and 8-ary 3-cube networks, respectively, using only 57% of the required silicon area for an adaptive wormhole router.

In summary, the adaptive Bubble router is a feasible alternative to existing commercial routers based on virtual channels and wormhole switching, such as SGI Spider, Cray T3E, and Intel Cavallino routers.

References

[1] A. Agarwal, "Limits on interconnection network performance," *IEEE Transactions on Parallel and Distributed Systems*, vol. 2, no. 4, pp. 398–412, October 1991.

[2] J. Carbonaro and F. Verhoorn, "Cavallino: The teraflops router and NIC," *Proceedings of Hot Interconnects Symposium IV*, August 1996.

[3] C. Carrión, R. Beivide, J.A. Gregorio and F. Vallejo, "A flow control mechanism to avoid message deadlock in k-ary n-cube networks," *Fourth International Conference on High Performance Computing*,pp. 322-329, India, December,1997.

[4] A.A. Chien, "A cost and speed model for k-ary n-cube wormhole routers," *Proceedings of Hot Interconnects'93*, Palo Alto, California, August 1993.

[5] R. Cypher and L. Gravano. "Storage-efficient deadlock-free packet routing algorithms for torus networks", *IEEE Trans. on Computers*, vol. C-43, no. 12, pp. 1376-1385, 1994.

[6] W. J. Dally and C. L. Seitz, "The torus routing chip," *Journal of Distributed Computing*, vol. 1, no. 3, pp. 187–196, October 1986.

[7] W.J. Dally and C.L. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. on Computers*, vol. C-36, no. 5, pp. 547-553, 1987.

[8] W. J. Dally, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Transactions on Computers*, vol. C–39, no. 6, pp. 775–785, June 1990.

[9] W.J. Dally, "Virtual-Channel Flow Control," *IEEE Trans. on Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194-205, March 1992.

[10] J. Duato,"A new theory of deadlock-free adaptive routing in wormhole networks." *IEEE Trans. on Parallel and Distributed Systems*, vol.4, no.12, pp.1320-1331, December 1993.

[11] J. Duato and P. Lopez,"Performance of adaptive routing algorithms k-ary n-cubes" *Proceedings of the Workshop on Parallel Computing Routing and Communications*, May 1994.

[12] J. Duato,"A necessary and sufficient condition for deadlock-free outing in cut-through and store-and-forward networks". *IEEE Trans. on Parallel and Distributed Systems*, vol.7, no.8, pp.841-854, August 1996.

[13] M. Galles, "Scalable pipelined interconnect for distributed endpoint routing: The SPIDER chip," *Proceedings of Hot Interconnects Symposium IV*, pp. 141-146, August 1996.

[14] P. Kermani and L. Kleinrock, "Virtual cut-through: a new computer communication switching technique,"*Computer Networks*, vol. 3, pp. 267-286, 1979.

[15] A. R. Larzelere II, "Creating simulation capabilities," *IEEE Computational Science & Engineering*, pp.27-35, January-March 1998.

[16] J. Laudon and D. Lenoski, "The SGI Origin: A cc-NUMA highly scalable server," *International Symposium on Computer Architecture*, pp. 241-251, June 1997.

[17] A. G. Nowatzyk and M. C. Browne and E. J. Kelly and M. Parkin "S-Connect: from NetWorks of Workstations to Supercomputer Peformance," *International Symposium on Computer Architecture*, pp. 71-82, June 1995.

[18] K. Bolding, M. Fulgham, and L. Snyder, "The Case of Chaotic Adaptive Routing", *IEEE Trans. on Computers*, vol. 46, no. 13, pp. 1281-1292, December 1997.

[19] S.L.Scott and G. Thorson, "The Cray T3E network: Adaptive routing in a high performance 3-D torus", *Hot Interconnects Symposium IV*, pp. 147-155, August 1996.

[20] S.L.Scott, "Synchronization and Communication in the T3E Multiprocessor", Proc. ASPLOS VII, Cambridge, MA, October 1996

[21] J.M. Prellezo, V. Puente, J.A. Gregorio and R. Beivide, "SICOSYS: an interconnection network simulator for parallel computers", Universidad de Cantabria Technical Report TR-ATC2-UC98, June 1998.

[22] V. Puente, J.A. Gregorio, C. Izu, R. Beivide and F. Vallejo "Low-level Router Design and its Impact on Supercomputer System Performance", *International Conference on Supercomputing*, June 1999.

[23] A.S. Vaidya, A. Sivasubramaniam and C.R. Das, "Performance benefits of virtual channels and adaptive routing: An application-driven study," *International Conference on Supercomputing*, pp. 140-147, July, 1997