The Reliable Router: A Reliable and High-Performance Communication Substrate for Parallel Computers [1]

William J. Dally, Larry R. Dennison, David Harris, Kinhong Kan and Thucydides Xanthopoulos.
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

1 Introduction

Interconnection networks play a major role in performance and reliability of massively parallel processors (MPPs). Previous work on network switching elements implementing oblivious routing such as the J-Machine Router [2], and the Caltech Mesh-Routing Chips [11] did not address the issue of reliability in part because of the inherent unreliability of oblivious routing. Past work on adaptive elements such as the Chaos Router [9] exploited adaptivity for performance reasons only without providing fault handling mechanisms. The Reliable Router (RR) uses adaptive routing coupled with link-level retransmission and a unique-token protocol to increase both performance and reliability. The RR can handle a single node or link failure *anywhere* in the network without interruption of service. Other unique features include a queueless low-latency plesiochronous channel interface, and simultaneous bidirectional signalling.

2 Architectural Overview

The Reliable Router is designed for two-dimensional mesh topologies. Its organization is shown in Figure 1. There is one Input Controller and one Output Controller for every direction. Moreover, there is a processor input/output and a diagnostic input/ output. Communication between an input and an output port occurs through a crossbar switch.

Most of the functionality of the chip has been pushed into the Input Controller. A simplified block diagram of the Input Controller is shown in Figure 2. Flits enter the module after being assembled and synchronized in the Front End and then are stored in a Flit Queue that corresponds to their virtual channel. If the flit is a head flit and its data payload contains routing information, it is fed through the Router which calculates a possible next step route depending on the destination address and on current availability of output virtual channels. The route is computed in terms of an Output Controller identifier (0-5) and a Virtual Channel identifier (0-4). This routing decision is not final and we call it "optimistic". If another Input Controller wants to route an incoming head flit to the same Output Controller, then the Crossbar Arbiter has to decide which one of the two head flits is allocated the resource.

[1] This research is supported by the Defense Advanced Research Projects Agency under contracts N00014-88K-0738, F19628-92-C-0045, and N00014-91-J-1698.

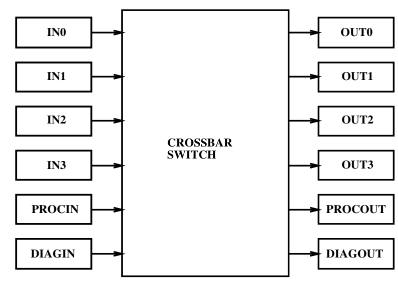

Figure 1: Organization of the Reliable Router

Every two cycles a decision must be made as to which Flit Queue – virtual channel – gets a chance to push a flit through the crossbar. A round robin scheduler picks among the eligible virtual channels. A virtual channel is considered eligible in two cases:

1. If the next flit of the channel is data or tail and the corresponding head flit of the packet has already gone through establishing the route. In this case, flow control – implemented by the Counter Bank – must also consent.
2. If the next flit is a head and the Router indicates that a possible next step route can be computed based on channel state information at that time.

The selection of the virtual channel to be popped is made by the Control box. When a flit is picked to be pushed through the crossbar, the new virtual channel identifier which is computed as part of the next step route is appended by the Flit Munger.

The Output Controllers simply append certain acknowledgment information, compute parity, and transmit the flit accross the link.

3 Adaptive Routing

Past research on adaptive routing has suffered from exponential dependence of resources on network dimension [10], carrying and updating message state along the route [3], and ad hoc fault-handling mechanisms incapable of handling faults along the edges of the network [10], [1]. The RR routing algorithm minimizes resource requirements and message state by using Duato's method [7]. The fault-handling properties of the routing algorithm are decoupled from the adaptive properties by using a different

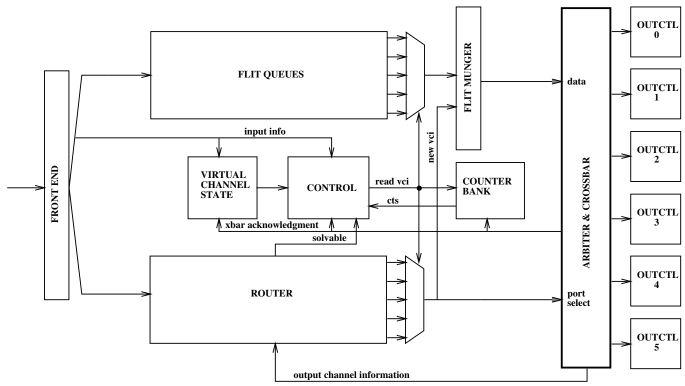

Figure 2: Input Controller Block Diagram

set of virtual channels and a different adaptive algorithm – the Turn Model [8] – for fault-handling. One can think of a network of RRs as the superposition of three separate virtual networks:

A minimally adaptive network. A packet using this network is able to route to any productive channel – a channel which will bring it closer to its destination. This virtual network tries to exploit adaptivity for performance reasons and is susceptible to deadlocks. The RR allocates two virtual channels to the adaptive network.

A dimension-ordered network. Packets in this network are routed in strict dimension order. Dimension-ordered routing is provably deadlock-free [4]. This virtual network exists in order to break deadlocks introduced in the previous network as suggested in [7]. The RR has two dimension-ordered virtual channels – one used by packets of priority 0 and one used by packets of priority one.

A fault-handling network. This network permits non-minimal adaptive steps and it is used to exploit the fault-handling properties of adaptive routing. We restrict the number of turns a packet can make and this network can be proved to be deadlock-free as suggested in [8]. The RR allocates one channel for fault-handling purposes.

All three virtual networks share the same physical network by using different sets of virtual channels. The routing algorithm consists of three separate computations occuring in parallel. The whole route computation takes a single clock cycle. Each computation results in a next step virtual channel from one of the three virtual networks described above:

Computation 1 The switching node attempts to find a non-busy minimally adaptive channel. If such a channel is found, then the message will use it as the next step of its path. Otherwise, the algorithm will pick the channel from Computation 2.

Computation 2 The packet is routed to the unique dimension-ordered channel corresponding to the current position and the destination of the packet. If this channel is busy, the packet blocks for one cycle and tries again to find a channel using Computation 1. The only case where Computation 2 fails to select the unique dimension-ordered channel is when that particular channel has gone down due to a fault. Only then do we select a channel from Computation 3.

Computation 3 The packet is routed to a fault-handling channel. This can be any channel, productive or unproductive, except for a channel that will cause the message to make a 180-degree turn. After the packet has been forwarded to the next switching node the algorithm either reverts to picking channels starting from Computation 1 or sticks to Computation 3 channels only and the packet reaches its destination through fault-handling channels. This decision depends on the packet dimension at the time Computation 2 fails to select a channel and is essential for the algorithm to be deadlock-free.

4 The Unique Token Protocol

End-to-end protocols may solve the reliability problem when coupled with adaptive routing, but require extra overhead and the necessary resources do not scale linearly with the size of the machine [5]. For this reason we decided to use link-level retransmission in combination with a unique-token protocol (UTP) [5] to guarantee fault-tolerant exactly-once delivery of all packets in the network. This link-level protocol offers significant advantages over end-to-end protocols because it does not require acknowledgment packets and does not keep copies for possible retransmissions at the packet source. In this way, effective network bandwidth is increased and storage requirements at the nodes decrease. Moreover, the protocol reduces the amount of storage required at the destination nodes for duplicate message detection. These properties allow the protocol resources to scale linearly

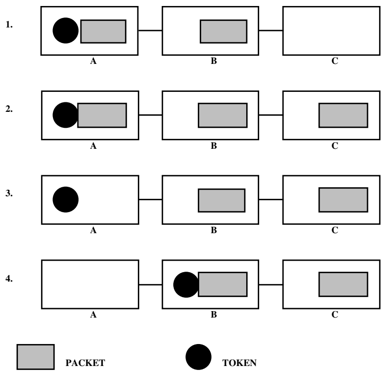

Figure 3: Buffering and forwarding under the UTP.

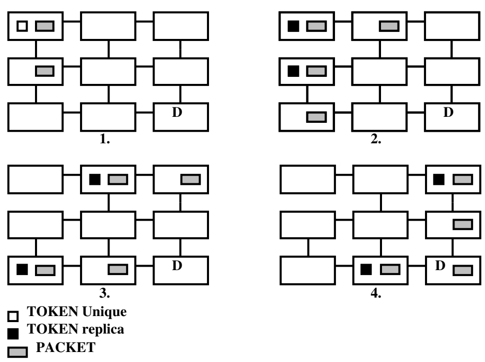

Figure 4: Fault Handling under the UTP

with the number of network nodes as opposed to end-to-end protocols.

An example of packet forwarding under the UTP is shown in Figure 3 where source node A sends a packet to destination node C. The packet is buffered and forwarded through switching node B. The process must ensure that at least two copies exist in the path between the source node and the destination node at all times. This can be achieved by first copying the packet forward one node, and then allowing the release of the storage in the rear-most node as shown in Figure 3. When the packet is first injected into the network, a token is injected right behind the packet. The invariant that no copies of the packet exist behind the token is always preserved. Packet copying and token passing is carried out exactly as shown in Figure 3. *Note that although multiple copies of the packet are kept in the network, every node receives a packet only once.* Thus, in the absence of faults, the arrival of the token at the destination implies that the packet has been delivered exactly once.

When a node in the network fails, communication between the advance and rear copies of the packet may be severed. Each copy now must make its way to the destination without knowing the fate of the other copy. When packets arrive at the destination they must be marked in such a way so that the destination knows that it needs to look for duplicates. For this reason we establish two types of tokens: Unique, and replica. If the network needs to use multiple paths while forwarding the packet, the token is changed to type replica for all copies of the packet. When the token is changed, forwarding proceeds in the usual way of always keeping two copies of the packet per path. Such an example is shown in Figure 4. Due to a faulty link, communication between the two copies of the packet has been broken. As a result, each copy changes its token to replica, or generates a replica token and proceeds to the destination using different paths. When the destination receives a packet with a replica token, it knows that it should be looking for duplicates. This scheme is based on the assumption that packets have unique identifiers so that duplicates can be detected and eliminated. For simplicity, the UTP was described at the packet level only. The Reliable Router implements the UTP at the flit level, and the protocol is a bit more involved.

5 Plesiochronous Data Recovery

One of the more difficult tasks in putting together a large scale MPP has been the global clock distribution. The RR addresses this problem by using plesiochronous timing. Each router is clocked using a different local oscillator with the same nominal frequency. There is no single global clock – and associated single point of failure – and the clock distribution problem is entirely avoided.

In a system of Reliable Routers, the clocks are all free running. To move data bits, the transmitter sends the clock along with the data. The receiver uses this clock to sample the data wires. Two latches are used to move data from the transmit clock domain to the receive clock domain. However, without some additional protocol, these latches will very quickly undersample or oversample flits.

We use a low-level protocol that imposes a maximum data rate on any wire which is below the minimum carrier rate of any wire in the system. The minimum carrier rate is determined by the lowest actual frequency of any of the local oscillators. The transmit limit is implemented in the router by turning off the crossbar one out of a thousand times. The Output Controllers send a non-data frame called a *padding flit* whenever there is no data to send.

The receiver performs FIFO adjustments depending on the overrun/underrun case. When tending toward overrun, the receiver does not enter padding flits into the FIFO. When tending toward underrun, the receiver inserts extra padding flits into the FIFO. The Input Controller simply ignores all padding flits.

This protocol, if implemented in the naive fashion, would occupy large amounts of area for the FIFO and add considerable latency. We have implemented it using two ranks of latches and a two-state state machine. The latency penalty is about 1.5 baud.

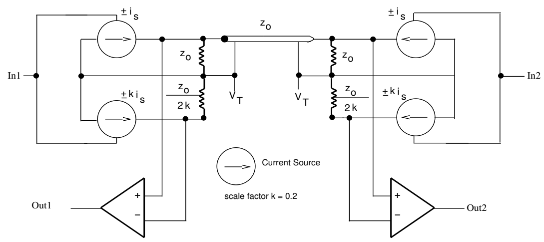

Figure 5: Simultaneous Bidirectional Signalling

6 Bidirectional Signalling

The 6 input and output ports shown in Figure 1 require a large amount of interchip bandwidth. To allow the use of a conventional chip carrier with fewer than 300 pins, we employ simultaneous bidirectional signalling [6]. This method allows bidirectional point to point digital signal communication on the same chip carrier pin and at the same time.

Figure 5 shows the block diagram of a pair of simultaneous bidirectional transceivers. The printed circuit board connection is modelled as a transmission line, with both ends of the line terminated with an internal resistor. A logic 1 or a logic 0 is transmitted as a positive current or a negative current respectively; the signal across the chip boundary is thus a superposition of the two current streams. To receive the digital signal from another transmitter, a local transmitter generates an internal copy of its own transmitted signal as a reference. The reference voltage is then subtracted from the superimposed signal to generate the received signal.

The small voltage swing (i.e. 250mV) of the signalling method calls for noise reduction techniques such as on-chip termination control, current steering and rise-time control. The current source for generating the local reference voltage, on the other hand, is scaled down to reduce power dissipation.

7 Conclusion

We have introduced the Reliable Router, a network switching element which provides reliable and high performance communication between nodes of parallel computers. The Reliable Router uses a simple and efficient routing algorithm with minimal resource requirements. It also employs a link-oriented retransmission protocol with significant advantages over conventional end-to-end protocols. The coupling of these two features enable the RR to handle a single node or link failure without interruption of service. Other performance features include a queueless low-latency plesiochronous channel interface and simultaneous bidirectional signalling.

The Reliable Router is designed to run internally at 100 MHz, with the links clocked at 200 MHz. It can achieve a maximum useful bandwidth of 3.2 Gbit/sec per link per direction.

We have full schematics for the system along with layout for most key modules. We are currently in the testing stage and will ship the design for fabrication in the beginning of April.

References

[1] Andrew A. Chien and Jae H. Kim. Planar Adaptive Routing: Low-cost Adaptive Networks for Multiprocessors. In *Proceedings of the 19th International Symposium on Computer Architecture*, pages 268–277, May 1992.

[2] William J. Dally et al. The Message-Driven Processor: A Multicomputer Processing Node with Efficient Mechanisms. *IEEE Micro*, April 1992.

[3] William J. Dally and Hiromichi Aoki. Deadlock-Free Adaptive Routing in Multicomputer Networks using Virtual Channels. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 4, No.4, April 1993.

[4] William J. Dally and Charles L. Seitz. Deadlock Free Routing in Multiprocessor Interconnection Networks. *IEEE Transactions on Computers*, C-36(5):547–53, May 1987.

[5] Larry R. Dennison. Reliable Interconnection Networks for Parallel Computers. MIT AI Laboratory Technical Report 1294, October 1991.

[6] Larry R. Dennison, Whay S. Lee and William J. Dally. High Performance Bidirectional Signalling in VLSI Systems. In *Research on Integrated Systems: Proceedings of the 1993 Symposium.*

[7] Jose Duato. On the Design of Deadlock-Free Adaptive Routing Algorithms for Multicomputers: Design Methodologies. In *Parallel Architectures and Languages Europe Proceedings*, pages 390-405, June 1991.

[8] Cristopher J. Glass and Lionel M. Ni. The Turn Model for Adaptive Routing. In *Proceedings of the 19th International Symposium on Computer Architecture*, pages 278–287, May 1992.

[9] S. Konstantinidou and L. Snyder. Chaos Router: Architecture and Performance. In *Proceedings of the 18th International Symposium on Computer Architecture*, pages 212–221, May 1991.

[10] Daniel H. Linder and Jim C. Harden. An Adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes. *IEEE Transactions on Computers*, C-40(1):2–12, January 1991.

[11] Charles L. Seitz and Wen-King Su. A Family of Routing and Communication Chips Based on the Mosaic. In *Research on Integrated Systems: Proceedings of the 1993 Symposium.*