The J-Machine Network

Peter R. Nuth   William J. Dally

Artificial Intelligence Lab,
Massachusetts Institute of Technology
Cambridge, MA 02139.

Abstract

The J-Machine network is a 3-D mesh employing wormhole routing and virtual channels to provide two network priorities. Each network channel is 9 bits wide and operates at 32MHz. Each J-Machine node contains network routers that guide messages between the six bidirectional channels incident on each node, and a network interface that handles messages originating or terminating at the node. The router is fully synchronous, uses absolute addressing, and performs dimension-order routing. A novel pad design permits the bidirectional channels to reverse direction each cycle without danger of conduction overlap.

1.0 Introduction

The J-Machine network is a 3-D mesh that provides a low latency communication fabric for up to 64K Message Driven Processor nodes. An MDP node is connected to each of its six nearest neighbors by a bidirectional channel transmitting data at 36 MB/sec. All message routing and delivery is performed autonomously by the network logic, requiring no processor intervention. The J-Machine network contains two *virtual channels* [3] that share the same physical wires, to support two logically independent message priorities.

In most large-scale multicomputers [1,5], composing and sending a message is an expensive operation, requiring 25µs to 500µs. In contrast, the MDP's on-chip network interface is tightly integrated with the processor core. An MDP transmits a message using a series of `SEND` instructions each of which injects one or two words into the network either at priority 0 or 1. Composing and injecting a four word message takes only 4 clock cycles (250ns). An injected message is delivered to the destination node by the network in ~1.5µs. At the destination, it is buffered in a hardware-managed FIFO queue in the internal RAM of the MDP. Separate queues are maintained for priority 0 and 1. The MDP hardware then signals the processor to call the appropriate handler code. The MDP creates a new task for each message and dispatches in <1µs.

2.0 Network Structure

The J-machine network is a three dimensional grid, with bidirectional channels, dimension-order routing, and blocking flow control. (See Figure 1). The 3-D network provides the highest throughput and lowest latency for a given wire density [2]. This network makes use of all three dimensions of physical space. It allows the processing nodes to be packed densely and results in uniformly short wires.

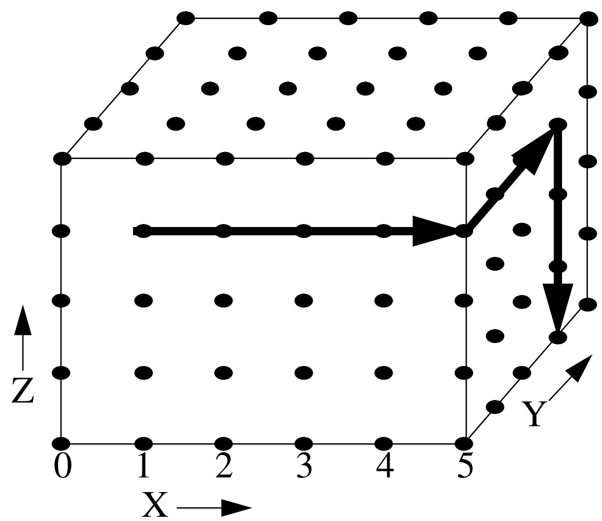

FIGURE 1. The J-Machine Network is a 3-D mesh or *k*-ary 3-cube. The network performs deterministic, dimension order routing. Messages are routed first in X, then Y, then Z. Here a message is routed from (1,0,3) to (5,3,0).

Chip pinout limits the width of each of the MDP's six bidirectional channels to 9 data bits plus 6 control bits. Node addressing limits the size of the network to 64K nodes (32×32×64). Our initial prototype is a 1K node machine (8×8×16). The faces of the network cube are left open for use as I/O ports. Each channel can sustain a data-rate of 288Mbits/sec. The bisection bandwidth[1] of a 1K-node machine is 18.4Gbits/s, the aggregate bandwidth of the network channels is 864Gbits/s, and the I/O bandwidth is 184Gbits/s.

The MDP router is synchronous[2] and operates at twice the rate of the processor sending a pair of 9-bit *phits* between nodes each 62.5ns processor cycle[3]. Each of the six bidirectional channels can be turned around on alternate cycles with no contention penalty. Messages are routed through the network with a latency of one processor cycle per hop. Thus message latency T is given by

$$T = T_c(2L + D), \quad \textbf{(EQ 1)}$$

where T_c is the processor cycle time, L is message length in words, and D is the distance (number of nodes) a message must traverse. For example, in a 1K-node machine, an L=6 word message to a random destination traverses an average of D=10 nodes for a latency of T=22 cycles or 1.4µs.

2.1 Routing & Flow control

The J-machine uses deterministic dimension-order routing (also called *e-cube* routing). As shown in Figure 1, all messages are first routed in the X dimension, then in Y, then in Z. Since messages are routed in dimension order and messages running in opposite directions along the same dimension do not block, there are no resource cycles, and the network is provably deadlock free [3].

Flit	Contents	Remark
1	5:+	X address
2	1:-	Y address
3	4:+	Z address
4	MSG:00	Method
5	00440	to call
6	INT:00	Argument
7	00023	to method
8	INT:00	Reply
9	<1:5:2> T	address

FIGURE 2. A typical message in the J-machine. The first three flits contain the destination address. The final flit in the msg is marked as the "tail".

Figure 2 shows the format of a message. The first three flits of the message contain the X, Y and Z addresses. At each node along the path, the address in the head flit of the message is compared with the node's index in the current dimension. If the two indices match, the head flit is

1. The bandwidth across a plane dividing the machine into two equal halves.

2. This breaks with a tradition of asynchronous network routers [4].

3. A *phit* is a physical digit, the width of the physical channel. A pair of phits form a *flit*, or flow-control digit, the granularity of flow-control in the network. An 18 bit flit is half an MDP data word.

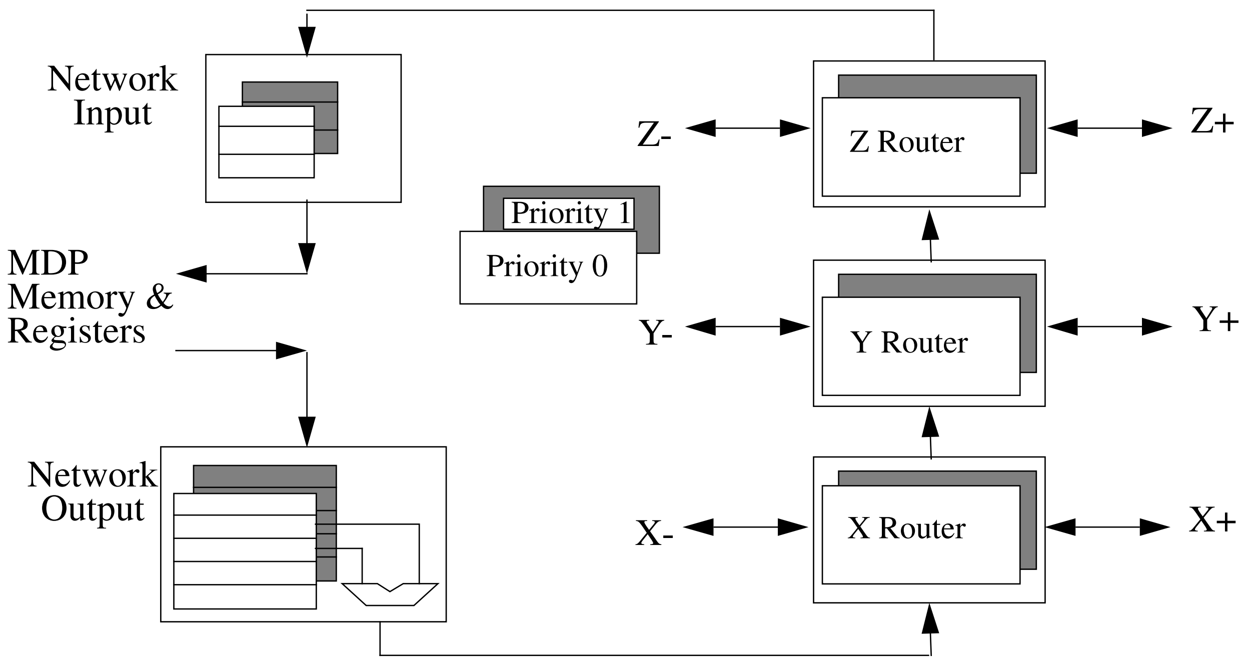

FIGURE 3. The MDP network is composed of a network output interface to inject messages into the network, three router modules to direct messages through the network, and a network input interface to reassemble messages destined for this node.

stripped off the message, and the rest is routed down to the next dimension. The address is formatted by the network output interface, which also precomputes the direction the message will take in each dimension. Setting these direction bits, (positive, negative or fall-through to next dimension) in the header words reduces the latency and complexity of the router nodes.

The network uses blocking flow control to resolve contention for a physical channel. When a message arrives at a router path that is already in use by a message of the same priority, it is blocked. The blocked message compresses into routers along its path, occupying one node per word (2 flits) of the message. When the blockage clears, the message uncompresses and proceeds to its destination, at a rate of one hop per cycle.

3.0 Implementation

The network logic is composed of three modules as shown in Figure 3. The network output module buffers words and injects them into the network. The three routers, one for each dimension of the network, route messages from node to node. The network input module reassembles messages at their destination and buffers them into a message queue.

3.1 Network Output

The network output block buffers message words from the MDP registers or memory and injects them into the network. It also formats the message address, and precomputes the direction the message will take in each network dimension. An eight word FIFO buffer matches the speed of message injection (2 words per cycle) to the network transmission rate (1/2 word per cycle). On each `SEND` instruction, the MDP transfers one or two words from processor registers or memory to the network output FIFO. When the message is complete, or the FIFO is half full, the message is launched into the network. In cases where the MDP cannot send message words as fast as the network can transmit them, the FIFO prevents *bubbles* (absence of words) from entering the network pipeline and degrading performance.

3.2 Network Input

The network input block transfers messages from the network to the MDP's memory. Data from the network arrives in 18-bit flits, which are composed into a four word *Queue Row Buffer* (QRB). When the QRB fills, its contents are written to the on-chip memory in one cycle. Writing memory a row (4×36 bits) at a time reduces the number of memory cycles consumed by the network, leaving more memory bandwidth for the CPU.

3.3 Network Routers

The routers form the switches in a J-Machine network and deliver messages to their destinations. As shown in Figure 3, the MDP contains three independent routers, one for each bidirectional dimension of the network. Each router is split into two separate virtual networks with different priorities that share the same physical channels. The priority 1 network can preempt the wires even if the priority 0 network is congested or jammed. Without this ability, there would be no way to redistribute data that has caused hot-spots in the network.

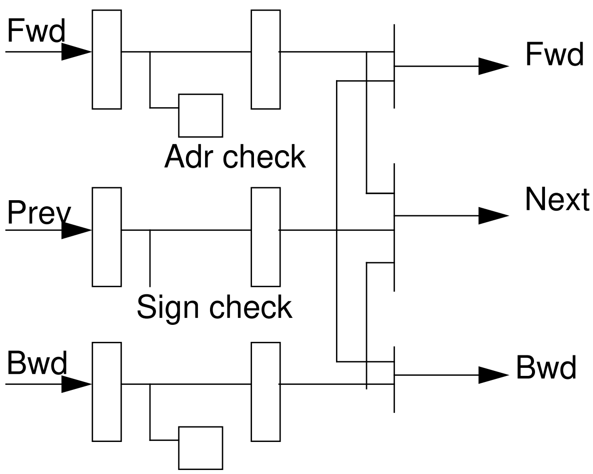

FIGURE 4. Each router priority contains forward, reverse, and previous to next dimension datapaths.

Each of the 18 router paths (Figure 4) contains buffering, comparators, and output arbitration. On each datapath, a comparator compares the lead flit, which contains the destination's address in this dimension, to the node coordinate. If the head flit does not match, the message continues in the current direction. Otherwise the message is routed to the next dimension. Messages entering the dimension compete with messages continuing in the dimension at a 2-to-1 switch. Once one message is granted this switch, the other input is locked out for the duration of the message. Once the head flit of the message has set up the route, subsequent flits follow directly behind it.

3.4 Pads

The MDP routers use a novel Input/Output pad to help compensate for clock skew in a large synchronous network. (See Figure 5). It is a standard I/O pad with the addition of a sustaining inverter *SINV*. When the pad's output enable signal *OEN* is turned on, the output driver easily overpowers the sustainer to drive a new logic value out of the pad. After *OEN* is dissasserted, the sustainer will hold the pad's current output value indefinitely.

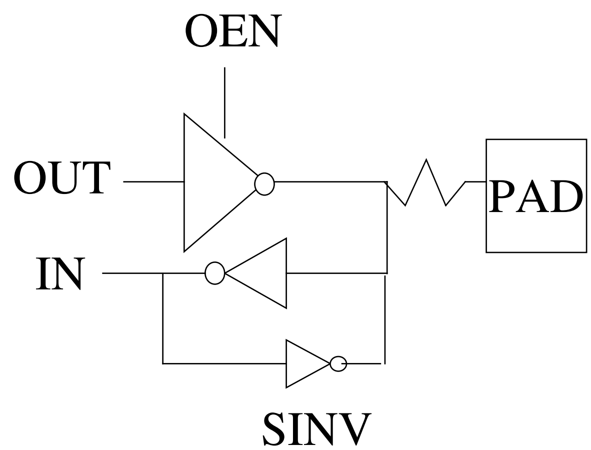

FIGURE 5. Schematic of an MDP I/O pad. Inverter *SINV* is a sustainer, to hold the current value on the pad output after output enable has been turned off.

For input-only pads, *SINV* is tied low to drive a logic low level on the input pin. This allows an MDP to be installed in systems without externally terminating unused network channels. Since the J-Machine mesh is open on all six sides, this internal termination simplifies processor board design.

3.5 Timing

The routers on either side of a bidirectional channel arbitrate for use of the channel. If both have data to send, one side will send a pair of 9 bit *phits*, then the opposing side will send its pair. Arbitration is pipelined with data transfer – the channel is never idle when there is data to send.

As shown in Figure 6, the MDPs are clocked by an external 32MHz *MCLK*. This is divided in half internally to produce a 16MHz processor clock. The routers drive net-

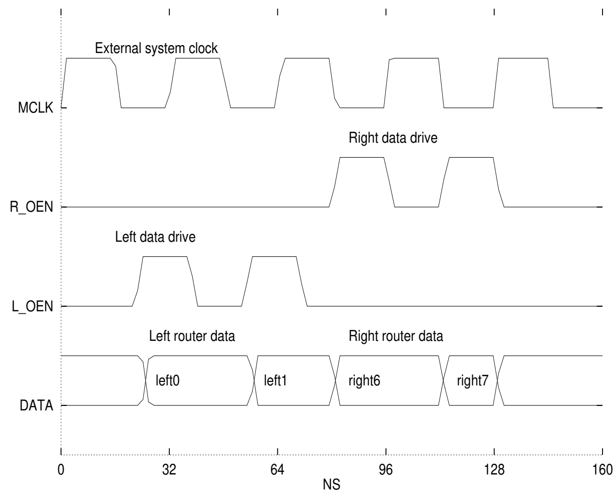

FIGURE 6. Router pad timing. Router signals are driven synchronous with external clock *MCLK*. Opposing routers trade use of the channel every other cycle.

work data once per *MCLK* (twice per processor cycle), gated by *MCLK* low. The sustaining pads then maintain that router data through the next cycle. The network signals are all latched by internal clocks *PHI1* and *PHI2*.

This clocking scheme serves two purposes: First, since the router pads sustain data until half-way through the next cycle, the channel can tolerate substantial internal and external clock skew between adjacent routers. Trading off channel speed for reliability eases the requirements on clock skew across a 4096 node J-Machine.

Second, the sustaining pads allow the channel to reverse direction every other cycle without two routers driving at the same time. This eliminates any fights between opposing routers pads and greatly reduces the power consumption of the routers.

4.0 Conclusion

The J-Machine network provides low latency communication among up to 64K processing nodes. It is tightly integrated with the MDP processor core for efficient message injection and reception. The network uses a deterministic, blocking routing strategy, to keep routing nodes fast and simple. The routers use a novel I/O pad design to help compensate for clock skew across channels, and to prevent fights on channel data lines.

Acknowledgements

The research described in this paper was supported in part by the Defense Advanced Research Projects Agency under contracts N00014-88K-0738 and N00014-91-J-1698, in part by a National Science Foundation Presidential Young Investigator Award, grant MIP-8657531, with matching funds from General Electric Corporation, IBM Corporation and AT&T Corporation, and in part by assistance from Intel Corporation.

References

1. Ramune Arlauskas. iPSC/2 System: A Second Generation Hypercube. In *Third Conference on Hypercube Concurrent Computers and Applications*, pages 33-36. ACM, 1988.
2. William J. Dally. Performance Analysis of *k*-ary *n*-cube Interconnection Networks. *IEEE Transactions on Computers*, 39(6), June 1990.
3. William J. Dally and Charles L. Seitz. Deadlock Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Transactions on Computers*, C-36(5), pages 547-53, May 1987.
4. William J. Dally and Paul Song. Design of a Self-Timed VLSI Multicomputer Communication Controller. In *Proceedings of the International Conference on Computer Design*, pages 230-234. IEEE, October 1987.
5. John F. Palmer. The NCUBE Family of Parallel Supercomputers. In *Proc. IEEE International Conference on Computer Design*, page 107. IEEE, 1986.