Express Cubes: Improving the Performance of k-ary n-cube Interconnection Networks

William J. Dally, *Member, IEEE*

***Abstract*—Express cubes are k-ary n-cube interconnection networks augmented by *express channels* that provide a short path for nonlocal messages. An express cube combines the logarithmic diameter of a multistage network with the wire-efficiency and ability to exploit locality of a low-dimensional mesh network. The insertion of express channels reduces the network diameter and thus the distance component of network latency. Wire length is increased allowing networks to operate with latencies that approach the physical speed-of-light limitation rather than being limited by node delays. Express channels increase wire bisection in a manner that allows the bisection to be controlled independent of the choice of radix, dimension, and channel width. By increasing wire bisection to saturate the available wiring media, throughput can be substantially increased. With an express cube both latency and throughput are wire-limited and within a small factor of the physical limit on performance. Express channels may be inserted into existing interconnection networks using *interchanges*. No changes to the local communication controllers are required.**

***Index Terms*—Communication networks, concurrent computing, interconnection networks, multicomputers, packet routing, packet switching, parallel processing, topology.**

I. Introduction

INTERCONNECTION networks are used to pass messages containing data and synchronization information between the nodes of concurrent computers [1], [2], [18], [19]. The messages may be sent between the processing nodes of a message-passing multicomputer [1] or between the processors and memories of a shared-memory multiprocessor [2].

An interconnection network is characterized by its topology, routing, and flow control [10]. The topology of a network is the arrangement of its nodes and channels into a graph. Routing determines the path chosen by a message in this graph. Flow control deals with the allocation of channel and buffer resources to a message as it travels along this path. This paper deals only with topology. Express cubes can be applied independent of routing and flow control strategies.

The performance of a network is measured in terms of its *latency* and its *throughput*. The latency of a message is the elapsed time from when the message send is initiated until the message is completely received. Network latency is the average message latency under specified conditions. Network throughput is the number of messages the network can deliver per unit time.

Low-dimensional k-ary n-cube networks using *wormhole routing* have been shown to provide low latency and high throughput for networks that are wire-limited [4], [5], [9]. For $n \leq 3$, the k-ary n-cube topology is wire-efficient in that it makes efficient use of the available bisection width. This topology maps into the three physical dimensions in a manner that allows messages to use all of the available bandwidth along their path without ever having to double back on themselves. Also, low-dimensional k-ary n-cubes concentrate bandwidth into a few wide channels so that the component of latency due to message length is reduced. In most contemporary concurrent computers, this is the dominant component of latency. Because of their low-latency, high throughput, and affinity for implementation in VLSI, these k-ary n-cube networks with $n = 2$ or 3 have been used successfully in the design of several concurrent computers including the Ametek 2010 [19], the J-Machine [7], [8], and the Mosaic [20].

However, low-dimensional k-ary n-cube interconnection networks have two significant shortcomings:

- Because wires are short, node delays dominate wire delays and the distance related component of latency falls more than an order of magnitude short of speed-of-light limitations. In the J-Machine [7], for example, node delay is 50 ns while the longest wire is 225 mm and has a time-of-flight delay of 1.5 ns.
- The channel width of these networks is often limited by node pin count rather than by wire bisection. For example, the J-Machine channel width is limited to 9-bits by pin count limitations. In the physical node width of 50 mm, a six-layer printed circuit board can handle over four times this channel width after accounting for through holes and local connections.

In short, many regular k-ary n-cube interconnection networks are node-limited rather than wire-limited. In these networks, node delay and pin limitations dominate wire delay and wire density limitations. The ratios of node delay to wire delays and pin density to wire density cannot be balanced in a regular k-ary n-cube.

Express cubes overcome this problem by allowing wire length and wire density to be adjusted independently of the choice of radix k, dimension n, and channel width W. An express cube is a k-ary n-cube augmented by one or more levels of express channels that allow nonlocal messages to

Manuscript received October 14, 1989; revised April 27, 1990. This work was supported in part by the Defense Advanced Research Projects Agency under Contracts N00014-88K-0738 and N00014-87K-0825 and in part by a National Science Foundation Presidential Young Investigator Award, Grant MIP-8657531, with matching funds from General Electric Corporation and IBM Corporation.

The author is with the Artificial Intelligence Laboratory and the Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139.

IEEE Log Number 9101688.

bypass nodes. The wire length of the express channels can be increased to the point that wire delays dominate node delays. The number of express channels can be adjusted to increase throughput until the available wiring media is saturated. This ability to balance node and wire limitations is achieved without sacrificing the wire-efficiency of k-ary n-cube networks. The number of channels traversed by a message in a hierarchical express cube grows logarithmically with distance as in a multistage interconnection network [12], [21]. The express cube, however, is able to exploit locality while in a multistage network all messages must traverse the diameter of the network. With an express cube, both latency and throughput are wire-limited and are within a small constant factor of the physical limit on performance.

The remainder of this paper describes the express cube topology and analyzes its performance. Section II summarizes the notation that will be used throughout the paper. Section III introduces the express cube topology in steps. Basic express cubes (Section III-A) reduce latency to twice the delay of dedicated wire for messages traveling long distances. Throughput can be increased to saturate the available wiring density by adding multiple express channels (Section III-B). With a hierarchical express cube (Section III-C), latency for short distances, while node-limited, is within a small constant factor of the best that can be achieved by any bounded degree network. Some design considerations for express cube interchanges are discussed in Section IV.

II. Notation

The following symbols are used in this paper. They are listed here for reference.

- C, the set of channels in the network.
- D, Manhattan distance traveled by a message, $|x_s - x_d| + |y_s - y_d| + |z_s - z_d|$, where the source is at (x_s, y_s, z_s) and the destination is at (x_d, y_d, z_d).
- f_j the fraction of traffic at level j in a hierarchical express cube.
- H *hops*, the number of nodes traversed by a message.
- i, number of nodes between interchanges in an express cube.
- k, the radix of the network—the length in each dimension.
- l, the number of levels of hierarchy in a hierarchical express cube.
- L, the message length in bits.
- m_j, the number of multiple express channels at level j.
- M, the number of express channels through each node.
- n, the dimension of the network.
- N, the set of nodes in the network. Where it is unambiguous, N is also used for the number of nodes in the network, $|N|$.
- T_n, the latency of a node.
- T_w, the latency of a wire that connects two physically adjacent nodes.
- T_p, the pipeline period of a node.
- W, the width of a channel in bits.
- $\mathcal{W}$, the width of a node—the number of wires that may pass into a node in each dimension.
- α, the ratio of node latency to wire latency, T_n/T_w.
- β, the ratio of channel width to node width, $\mathcal{W}/W$.

An interconnection network consists of a set of nodes N that are connected by a set of channels, $C \subseteq N \times N$. Each channel is unidirectional and carries data from a source node to a destination node. For the purposes of this paper it is assumed that the network is bidirectional: channels occur in pairs so that $(n_1, n_2) \in C \Rightarrow (n_2, n_1) \in C$.

Communication between nodes is performed by sending *messages*. A message may be broken into one or more *packets* for transmission. A packet is the smallest unit that contains routing and sequencing information. Packets contain one or more flow control digits or *flits*. A flit is the smallest unit on which flow control is performed. A flit in turn is composed of one or more physical transfer units or *phits*.[1] A phit is W-bits, the size of the physical communication media.

The express cube topology is particularly suitable for use with wormhole routing, a flow-control protocol that advances each flit of a packet as soon as it arrives at a node (pipelining) and blocks packets in place when required resources are unavailable [4], [5], [9]. Wormhole routing is attractive in that 1) it reduces the latency of message delivery compared to store and forward routing, and 2) it requires only a few flit buffers per node. Wormhole routing differs from virtual cut-through routing [11] in that with wormhole routing it is not necessary for a node to allocate an entire packet buffer before accepting each packet. This distinction reduces the amount of buffering required on each node making it possible to build fast, inexpensive routers.

The *bisection width* of a network is the minimum number of channels that must be cut to partition the network into two equal halves. The *wire bisection* is the number of wires in this channel cutset. Bisection width gives a lower bound on *wire density*, the maximum number of wires that must cross a unit distance (2-D) or area (3-D).

III. Express Cubes

A. Express Channels Reduce Latency

Fig. 1 illustrates the application of express channels to a k-ary 1-cube or linear array. A regular k-ary 1-cube is shown in Fig. 1(a). The network is a linear array of k processing nodes, labeled N, each connected to its nearest neighbors by channels of width W. The delay of a phit propagating through a node is T_n. The delay of the wire connecting two nodes is T_w. Each channel can accept a new phit every T_p. The latency of a message of length L sent distance D is

$$T_a = HT_n + DT_w + \frac{L}{W}T_p. \tag{1}$$

[1]There is no constraint that the physical unit of transfer, phit, must be smaller than the flow control unit, flit. It is possible to construct systems with several flits in each phit.

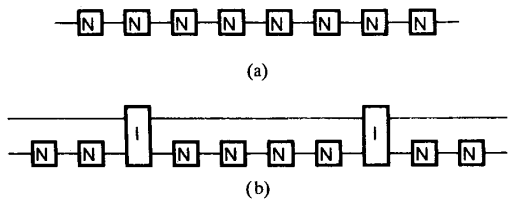


Fig. 1. Insertion of express channels reduces latency: (a) A regular k-ary 1-cube network may be dominated by node delay. (b) A k-ary 1-cube with express channels reduces the node delay component of latency.

Message latency is composed of three components as shown in (1).[2] The first component is the node latency, due to the number of hops H. The second component is the wire latency, due to the distance D. The third component is due to message length L.

For a conventional k-ary n-cube, $H = D$ giving

$$T_a = (T_n + T_w)D + \frac{L}{W}T_p. \tag{2}$$

For most networks $T_n >> T_w$ so the node latency dominates the wire latency. Express cubes reduce the node latency by increasing wire length to reduce the number of hops H.

An express k-ary 1-cube is shown in Fig. 1(b). Express channels have been added to the array by inserting an interchange, labeled I, every i nodes. An interchange is not a processing node. It performs only communication functions and is not assigned an address. Each interchange is connected to its neighboring interchanges by an additional channel of width W, the express channel. When a message arrives at an interchange it is routed directly to the next interchange if it is not destined for one of the intervening nodes. To preserve the wire-efficiency of the network, messages are never routed past their destinations on the express channels even though doing so would reduce H in many cases.

The delay T_n, and throughput $1/T_p$, of an interchange are assumed to be identical to those of a node. The wire delay of the express channel is assumed to be iT_w. To simplify the following analysis, it is assumed that interchanges add no physical distance to the network. Assuming $i \mid D$, $H = D/i + i$ and insertion of express channels reduces the latency to

$$T_b = \left(\frac{D}{i} + i\right)T_n + T_w D + \frac{LT_p}{W}. \tag{3}$$

In the general case, $i \nmid D$, an average message traversing D processing nodes travels over $H_i = (i+1)/2$ local channels to reach an interchange, $H_e = \lfloor D/i - 1/2 + 1/(2i) \rfloor$ express channels to reach the last interchange before the destination, and finally $H_f = (1 + (D - i/2 - 1/2) \bmod i)$ local channels to the destination. The total number of hops is $H = H_i + H_e + H_f$ giving a latency of

$$T_b = \left(\frac{i+1}{2} + \left\lfloor \frac{D}{i} - \frac{1}{2} + \frac{1}{2i} \right\rfloor + \left(1 + \left(D - \frac{i}{2} - \frac{1}{2}\right) \bmod i\right)\right)T_n + DT_w + \frac{LT_p}{W}. \tag{4}$$

For large distances, $D >> \alpha = T_n/T_w$, choosing $i = \alpha$ balances the node and wire delay. With this choice of i, the latency due to distance is approximately twice the wire latency, $T_D \approx 2T_w D$. The latency for large distances of an express channel network with $i = \alpha$ is within a factor of two of the latency of a dedicated Manhattan wire between the source and destination.[3]

For small distances or large α, the i term in the coefficient of T_n in (3) is significant and node delay dominates. For such networks, latency is minimized by choosing $i = \sqrt{D}$ resulting in $T_D \approx 2(\sqrt{D} - 1)T_n$. The use of hierarchical express channels (Section III-C) can further improve the latency for small distances.

B. Multiple Express Channels Increase Throughput to Saturate Wire Density

To first order, network throughput is proportional to wire bisection and hence wire density. If more wires are available to transmit data across the network, throughput will be increased provided that routing and flow control strategies are able to profitably schedule traffic onto these wires. Many regular network topologies, such as low-dimensional k-ary n-cubes, are unable to make use of all available wire density because of pin limitations. The wire bisection of an express cube can be controlled independent of the choice of radix k, dimension n, or channel width W by adding multiple express channels to the network to match network throughput with the available wiring density $\mathcal{W}$.

Fig. 2 shows two methods of inserting multiple express channels. Multiple express channels may be handled by each interchange as shown in Fig. 2(a). Alternatively, simplex interchanges can be interleaved as shown in Fig. 2(b).

In method (a), using multiple channel interchanges, an interchange is inserted every i nodes as above and each interchange is connected to its neighbors using m parallel express channels. Fig. 2(a) shows a network with $i = 4$ and $m = 2$. The interchange acts as a concentrator combining messages arriving on the m incoming express channels with nonlocal messages arriving on the local channel and concentrating these messages streams onto the m outgoing express channels. This method has the advantage of making better use of the express channels since any message can route on any express channel. Flexibility in express channel assignment is achieved at the expense of higher pin count and limited expansion.

With method (b), interleaving simplex interchanges, m simplex interchanges are inserted into each group of i nodes. Each interchange is connected to the corresponding interchange in the next group by a single express channel. All messages from the nodes immediately before an interchange will be routed on that interchange's express channels. Because load cannot

[2] Throughout this paper the term latency is used to refer to the latency of a single message in the absence of traffic. For a discussion of the effects of traffic on latency see [5] and [9].

[3] There is nothing special about the factor of two. By choosing $i = j\alpha$ the distance component of latency will be $(1 + 1/j)$ times the latency of a Manhattan wire.

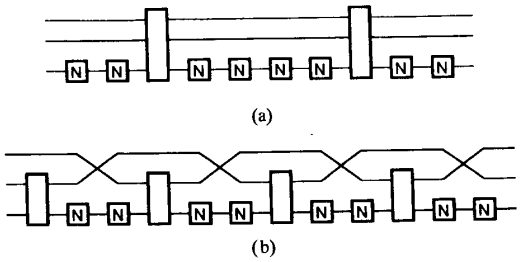


Fig. 2. Multiple express channels allow wire density to be increased to saturate the available wiring media. Express channels can be added using either (a) interchanges with multiple express channels, or (b) interleaved simplex interchanges.

be shared among interleaved express channels, an uneven distribution of traffic may result in some channels being saturated while parallel channels are idle. Method (b) has the advantage of using simple interchanges and allowing arbitrary expansion. In the extreme case of inserting an interchange between every pair of nodes the resulting topology is almost the same as the topology that would result from doubling the number of dimensions.

Both of the methods illustrated in Fig. 2 have the effect of increasing the wire density (and bisection) by a factor of $m + 1$. To first order, network throughput will increase by a similar amount. There will be some degradation due to uneven loading of parallel channels.

The use of multiple express channels offsets the load imbalance between express and local channels. If traffic is uniformly distributed, the average fraction of messages crossing a point in the center of the network on a local channel is $f_0 = 2i/k$ as compared to $f_1 = (k - 2i)/k$ crossing on an express channel. For large networks where $k >> i$, the bulk of the traffic is on express channels. Increasing the number of express channels applies more of the network bandwidth where it is most needed. The issue of allocating multiple express channels is discussed further in Section III-E.

Multiple express channels are an effective method of increasing throughput in networks where the channel width is limited by pinout constraints. For example, in the J-Machine the channel width $W = 9$ is set by pin limitations.[4] The printed-circuit board technology is capable of running $\mathcal{W} = 80$ wires in each dimension across the 50 mm width of a node. Even with many of these wires used for local connections, four parallel 15-bit (data + control) wide channels can be easily run across each node. A multiple express channel network with $m = 3$ could use this available wire density to quadruple the throughput of the network.

C. Hierarchical Express Cubes Have Logarithmic Node Delay

With a single level of express channels, an average of i local channels are traversed by each nonlocal message. The node delay on these local channels represents a significant component of latency and causes networks with short distances, $D \leq \alpha^2$, to be node limited. Hierarchical express cubes overcome this limitation by using several levels of express channels to make node delay increase logarithmically with distance for short distances.

The use of hierarchical express channels, shown in Fig. 3, reduces the latency due to node delay on local channels. With hierarchical express channels, there are l levels of interchanges. A first-level interchange is inserted every i nodes. A second-level interchange replaces every ith first level interchange, every i^2 nodes. In general, a jth level interchange replaces every ith $j-1$st level interchange, every i^j nodes.[5] Fig. 3 illustrates a hierarchical express cube with $i = 2$, $l = 2$.

A jth level interchange has $j + 1$ inputs and $j + 1$ outputs. Arriving messages are treated identically regardless of the input on which they arrive. Messages that are destined for one of the next i nodes are routed to the local (0th) output. Those remaining messages that are destined for one of the next i^2 nodes are routed to the 1st output. The process continues with all messages with a destination between i^p and i^{p+1} nodes away, $0 \leq p \leq j - 1$, routed to the pth output. All remaining messages are routed to the jth output.

A message in a hierarchical express cube is delivered in three phases: ascent, cruise, and descent. In the ascent phase, an average message travels $(i + 1)/2$ hops to get to the first interchange, and $(i - 1)/2$ hops at each level for a total of $H_a = (i - 1)/2 + 1$ hops and a distance of $D_a = (i^l - 1)/2$. During the cruise phase, a message travels $H_c = \lfloor (D - D_a)/i^l \rfloor$ hops on level l channels for a distance of $D_c = i^l H_c$. Finally, the message descends back through the levels routing on each level, j, as long as the remaining distance is greater than i^j. For the special case where $i^l \mid D$, the descending message takes $H_d = (i - 1)l/2 + 1$ hops for a distance of $D_d = (i^l + 1)/2$. This gives a latency of

$$T_c = \left(\frac{D}{i^l} + (i-1)\,l + 1 \right) T_n + T_w D + \frac{LT_p}{W}. \quad (5)$$

Choosing i and l so that $i^l = \alpha$ balances node and wire delay for large distances. With this choice, the delay due to local nodes is $(i - 1)\,l\,T_n = (i - 1)\,\log_i \alpha T_n$. Given that i is an integer greater than unity, this expression is minimized for $i = 2$. Choosing i to be a power of two facilitates decoding of binary addresses in interchanges. Networks with $i = 4$, $i = 8$, and $i = 16$ may be desirable under some circumstances.

In the general case, $i^l \nmid D$, the latency of a hierarchical express cube is calculated by representing the source and destination coordinates as $h = \log_i k$-digit radix-i numbers, $S = s_{h-1} \cdots s_0$, and $D = d_{h-1} \cdots d_0$. Without loss of generality we assume that $S < D$. During the ascent phase, a message routes from S to $s_{h-1} \cdots s_{l+1} 0 \cdots 0$ taking $H_a = \Sigma_{j=0}^{l-1} ((i - s_j) \bmod i)$ hops for a distance of $D_a = \Sigma_{j=0}^{l-1} ((i - s_j) \bmod i) i^j$. The cruise phase takes the message $H_c = \Sigma_{j=l}^{h-1} (d_j - s_j) i^{j-l}$ hops for a distance of $D_c = H_c i^l$. Finally, the descent phase takes the message from $d_{h-1} \cdots d_l 0 \cdots 0$ to D taking $H_d = \Sigma_{j=0}^{l-1} d_j$ hops for a distance of $D_d = \Sigma_{j=0}^{l-1} d_j i^j$.

[4] Each J-Machine node is packaged in a 168-pin pin-grid array. The six communication channels each require 9 data bits and six control bits consuming 90 of these pins. Power connections use 48 pins. The remaining 30 pins are used by external memory interface and control [16].

[5] This construction yields a fixed-radix express cube, with radix i for each level. It is also possible to construct mixed-radix express cubes where the radix varies from level to level.

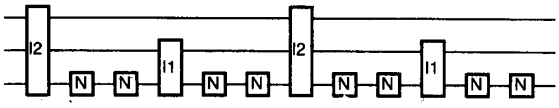


Fig. 3. Hierarchical express channels reduce latency due to local routing.

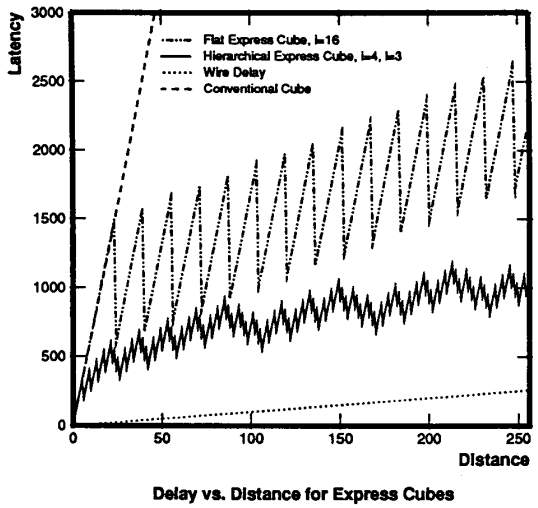


Fig. 4. Latency as a function of distance for a hierarchical express channel cube with $i = 4$, $l = 3$, $\alpha = 64$, and a flat express channel cube with $i = 16$, $\alpha = 64$. In a hierarchical express channel cube latency is logarithmic for short distances and linear for long distances. The crossover occurs between $D = \alpha$ and $D = i\alpha \log_i \alpha$. The flat cube has linear delay dominated by T_n for short distances and T_w for long distances.

For short distances the cruise phase will never be reached. The message will move from ascent to descent as soon as it reaches a node where all nonzero coordinates agree with D. The total latency for the general case is plotted as a function of distance in Fig. 4.

Fig. 5 shows how hierarchical interchanges can be implemented using pin-bounded modules. A level-j interchange requires $j + 1$ inputs and outputs if implemented as a single module as shown for a third level interchange in Fig. 5(a). A level-j interchange can be decomposed into $2j - 1$ level-one interchanges as shown for $j = 2$ in Fig. 5(b). A series of $j - 1$ ascending interchanges that route nonlocal traffic toward higher levels is followed by a top-level interchange and a series of $j - 1$ descending interchanges that allow local traffic to descend. With some degradation in performance, the ascending interchanges can be eliminated as shown in Fig. 5(c). This change requires extra hops in some cases as a message cannot skip levels on its way up to a high-level express channel. Each message must traverse at least one level $j - 1$ channel before being switched to a level-j channel. By restricting messages to also travel on at least one channel at each level as they descend, the descending interchanges can be eliminated as well leaving only the single top-level interchange as shown in Fig. 5(d).

D. Performance Comparison

Fig. 4 shows how latency varies with distance in hierarchical and flat express cubes and compares these latencies to the latency of a conventional k-ary 1-cube and of a direct wire. These curves assume that the message source is midway between two interchanges. The latencies are normalized to units of the wire delay between adjacent nodes. The latency of a conventional k-ary 1-cube is linear with slope α while the latency of a wire is linear with slope 1.

For short distances, until the first express channel is reached, a flat (nonhierarchical) express cube has the same delay as a conventional k-ary n-cube, $T_D = \alpha D$. Once the message begins traveling on express channels, latency increases linearly with slope $1 + \alpha/i$. This occurs at distance $D = 24$ in the figure. There is a periodic variation in delay around this asymptote due to the number of local channels being traversed, $D_{\text{local}} = (i + 1)/2 + ((D - i/2 + 1/2) \bmod i)$.

The hierarchical express cube has a latency that is logarithmic for short distances and linear for long distances. The latency of messages traveling a short distance, $D < \alpha$ is node limited and increases logarithmically with distance, $T_D \approx (i - 1) \log_i D T_n$. This delay is within a factor of $i - 1$ of the best that can be achieved with radix i switches. Long distance messages have a latency of $T_D \approx (1 + \alpha/i^l) T_w$. If $i^l = \alpha$, this long distance latency is approximately twice the latency of a dedicated Manhattan wire. In a hierarchical network, the interchange spacing i can be made small, giving good performance for short distances, without compromising the delay of long distance messages which depends on the ratio α/i^l. In a flat network with a single parameter i, it is not possible to simultaneously optimize performance for both short and long distances.

E. Area Tradeoffs

Assume that a node has a cross-sectional area that permits $\mathcal{W}$ wires to pass through in each dimension. W of these wires are used for a local channel. The remaining $\mathcal{W} - W$ wires are allocated as $M = \lfloor \beta - 1 \rfloor$ W-wire channels since a narrower channel will form a bottleneck that will slow other channels. The M available channels should be divided among the levels in a hierarchical express cube in a manner that evenly balances the load.

Assuming random traffic, at each level j, from $j = 0$, local channels, to $j = l - 1$, the fraction of traffic carried at level j on a channel near the center of the machine is

$$f_j = \frac{2i^j}{k}. \tag{6}$$

The fraction of traffic on the top-level channel is the remainder,

$$f_l = \frac{i^l - i}{1 - i}. \tag{7}$$

To balance the load between levels of the express cube, m_j channels should be allocated to each level j of the cube in proportion to the fraction of traffic f_j at that level. In practice, M is not large enough to permit an exact balance. For example, in a cube with $k = 64$, $i = 2$, $l = 3$, and random traffic, the fractions, f_0 to f_3, are 0.0625, 0.125, 0.250, and 0.5625,

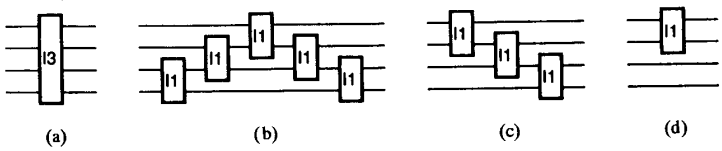


Fig. 5. Hierarchical interchanges. (a) A third-level interchange. (b) A third-level interchange implemented from first-level interchanges. (c), (d) With a small performance penalty, ascending and/or descending interchanges can be eliminated.

respectively. If $M = 7$, a reasonable allocation to m_0 through m_3 is 1, 1, 2, and 4, respectively.

If there is considerable locality in the traffic pattern, more traffic will travel on the lower levels and additional channels should be allocated to these levels. It is better to overallocate channels to lower levels than to higher levels because the lower-level channels are more versatile. Long distance messages can make use of lower level channels with some increase in latency; however, local messages cannot make progress on high-level channels.

F. Express Channels in Many Dimensions

A multidimensional express cube may be constructed by inserting interchanges into each dimension separately as shown in Fig. 6(a). The figure shows part of a two-dimensional express cube with $i = 4$, $l = 1$. Interchanges have been inserted separately into the X and Y dimensions. A similar construction can be realized for higher dimensions and for hierarchical networks. With this approach interchange pin-count is minimal as each interchange handles only a single dimension. Also, the design is easy to package into modules as the interchanges are located in regular rows and columns. This approach has the disadvantage that messages must descend to local channels to switch dimensions.

An alternate construction of a multidimensional express cube is to interleave multidimensional interchanges into the array as shown in Fig. 6(b) for $i = 4$, $l = 1$. This approach allows messages on express channels to change dimensions without descending to a local channel. It is particularly useful in networks that use adaptive routing [14], [15] as it provides alternate paths at each level of the network. The interleaved construction has the disadvantages of requiring a higher interchange pin count and being more difficult to package into modules.

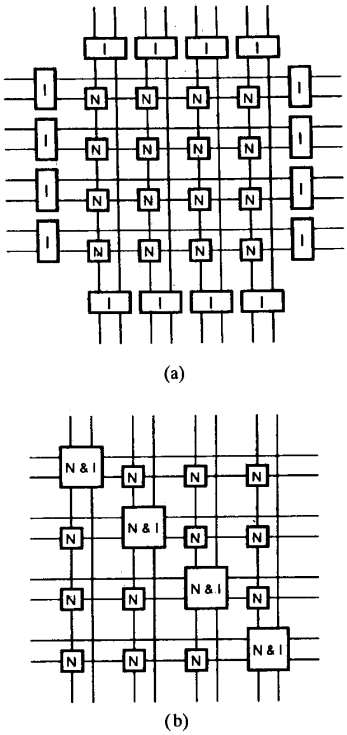


Fig. 6. A multidimensional express cube may be constructed either by (a) inserting interchanges into each dimension separately, or (b) interleaving multidimensional interchanges into the array.

G. Modularity

The interchanges in an express cube can be used to change wire density, speed, and signaling levels at module boundaries as shown in Fig. 7. Large networks are built from many modules in a physical hierarchy. A typical hierarchy includes integrated circuits, printed circuit boards, chassis, and cabinets. Available wire density and bandwidth change significantly between levels of the hierarchy. For example, a typical integrated circuit has a wire density of 250 wires/mm per layer while a printed circuit board can handle only 2 wires/mm per layer.[6] Interchanges placed at module boundaries as shown in Fig. 7 can be used to vary the number and width of express and local channels. These boundary interchanges may also convert internal module signaling levels and speeds to levels and speeds more appropriate between modules. Using express channels and boundary interchanges, the network can be adjusted to saturate the available wiring density even though this density is not uniform across the packaging hierarchy. To make use of the available bandwidth, computations running on the network must exploit locality.

[6]This integrated circuit wire density is typical of first-level metal in a 1 μm CMOS process. The printed circuit wire density is for a board with 8 mil wires and spaces. Both densities assume all area is available for wiring.

IV. Interchange Design

Fig. 8 shows the block diagram of a unidirectional interchange. A bidirectional interchange includes an identical circuit in the opposite direction. The basic design is similar to that of a router [17], [6], [3]. Two input latches hold arriving flits and two output latches hold departing flits. If additional buffering is desired, any of these latches may be replaced by a FIFO buffer. If a phit is a different size than a flit,

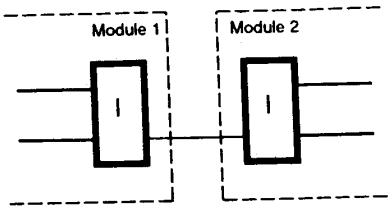


Fig. 7. Interchanges allow wire density, speed, and signaling levels to be changed at module boundaries.

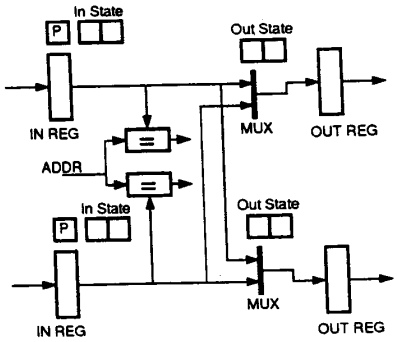


Fig. 8. Block diagram of an interchange. Two multiplexors perform switching between input and output registers based on a comparison of the high address bits in a message header.

multiplexing and demultiplexing is required between the flit buffers and the interchange pins. Associated with each output latch is a multiplexor that selects which input is routed to the latch. Routing decisions are made by comparing the address information in the head flit(s) of the message to the local address. If the destination lies within the next i nodes, the local channel is chosen, otherwise the express channel is chosen. If i is a power of two, interchanges are aligned, and absolute addresses are used in headers, the comparison can be made by checking all but the $l \log_2 i$ least significant bits for equality to the local address.

The interchange state includes presence bits for each register, an input state for each input, and an output state for each output. The presence bits are used for flit-level flow control. A flit is allowed to advance only if the presence bit of its destination register is clear (no data present), or if the register is to be emptied in the same cycle. The input state bits hold the destination port and status (empty, head, advancing, blocked) of the message currently using each input. The output state consists of a bit to identify whether the output is busy and a second bit to identify which input has been granted the output. The combinational logic to maintain these state bits and control the data path is straightforward.

V. Conclusion

Express cubes are k-ary n-cubes augmented by express channels that provide a short path for nonlocal messages. An express cube retains the wire efficiency of a conventional k-ary n-cube while providing improved latency and throughput that are limited only by the wire delay and available wire density. For short distances, a hierarchical express cube has a latency that is within a small factor of the best that can be achieved with a bounded degree network. For long distances, the latency can be made arbitrarily close to that of a dedicated Manhattan wire. Multiple express channels can be used to increase throughput to the limit of the available wire density. The express cube combines the low diameter of multistage interconnection networks with the wire efficiency and ability to exploit locality of a low-dimensional mesh network. The result is a network with latency and throughput that are within a small factor of the physical limit.

Express channels are added to a k-ary n-cube by periodically inserting interchanges into each dimension. No modifications are required to the routers in each processing node; express channels can be added to most existing k-ary n-cube networks. Interchanges also allow wire density, speed, and signaling levels to be changed at module boundaries. An express cube can make use of all available wire density even if the wire density is nonuniform. This is often required as the wire density and speed may change significantly between levels of packaging.

Express cubes achieve their performance at the cost of adding interchanges, increasing the latency for some short-distance messages, and increasing the bisection width of the network. Each interchange adds a component to the system and increases the latency of local messages that cross an interchange but do not take the express channel by one node delay, $(T_n + T_w)$. Express channels increase the wire bisection by using available unused wiring capacity. In parts of the network that are already wire-limited the express and local channels can be combined as shown in Fig. 7.

As the performance of interconnection networks approaches the limits of the underlying wiring media their range of application increases. These networks can go beyond exchanging messages between the nodes of concurrent computers to serving as a general interconnection media for digital electronic systems. For distances larger than $D' = \alpha i \log_i \alpha$, the delay of a hierarchical express cube network is within a factor of three of that of a dedicated wire. The network may provide better performance than the wire because it is able to share its wiring resources among many paths in the network while a dedicated wire serves only a single source and destination. For distances smaller than D', dedicated wiring offers a significant latency advantage at the cost of eliminating resource sharing.

Acknowledgment

I thank C. Leiserson for pointing out the node-limited nature of most real k-ary n-cubes. Express cubes have been strongly influenced by his work on fat trees [13]. I thank S. Ward, A. Agarwal, and T. Knight for many helpful comments and suggestions about routing networks and their analysis. I thank C. Seitz for many helpful suggestions about networks, routers, and concurrent computers. I thank A. Chien and M. Noakes for their careful review of early drafts of this manuscript. Finally I thank all the members of the MIT Concurrent VLSI

Architecture group for their help with and contributions to this paper.

REFERENCES

[1] W. C. Athas and C. L. Seitz, "Multicomputers: Message-passing concurrent computers," *IEEE Comput. Mag.*, vol. 21, pp. 9–24, Aug. 1988.

[2] BBN Advanced Computers, Inc., "Butterfly parallel processor overview," BBN Rep. 6148, Mar. 1986.

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

[4] W. J. Dally, *A VLSI Architecture for Concurrent Data Structures.* Hingham, MA: Kluwer, 1987.

[5] ——, "Wire efficient VLSI multiprocessor communication networks," in *Proc. Stanford Conf. Advanced Res. VLSI,* P. Losleben, Ed. Cambridge, MA: MIT Press, Mar. 1987, pp. 391–415.

[6] W. J. Dally and P. Song, "Design of a self-timed VLSI multicomputer communication controller," in *Proc. Int. Conf. Comput. Design, ICCD-87,* 1987, pp. 230–234.

[7] W. J. Dally *et al.*, "The J-Machine: A fine-grain concurrent computer," in *Proc. IFIP Congress,* 1989.

[8] W. J. Dally, "The J-Machine: System support for actors," in *Actors: Knowledge-Based Concurrent Computing,* Hewitt and Agha, Eds. Cambridge, MA: MIT Press, 1991.

[9] ——, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Trans. Comput.*, vol. 39, pp. 775-785, June 1990.

[10] ——, "Network and processor architecture for message-driven computing," in *VLSI and Parallel Processing,* R. Suaya and G. Birtwistle, Eds. Los Altos, CA: Morgan Kaufmann, 1990.

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

[12] D. H. Lawrie, "Alignment and access of data in an array processor," *IEEE Trans. Comput.*, vol. C-24, pp. 1145–1155, Dec. 1975.

[13] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Trans. Comput.*, vol. C-34, pp. 892–900, Oct. 1985.

[14] J. Mailhot, "A comparative study of routing and flow control strategies in k-ary n-cube networks," S.B. thesis, Massachusetts Instit. of Technol., May 1988.

[15] J. Ngai, "A framework for adaptive routing in multicomputer networks," Ph.D. dissertation, Caltech Computer Science Tech. Rep., Caltech-CS-TR-89-09, May 1989.

[16] M. O. Noakes and W. J. Dally, "System design of the J-Machine," in *Proc. Sixth MIT Conf. Advanced Res. VLSI,* MIT Press, 1990, pp. 179–194.

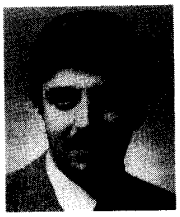
[17] P. R. Nuth, "Router protocol," MIT Concurrent VLSI Architecture Memo 23, Feb. 1989.

[18] C. L. Seitz, "The Cosmic Cube," *Commun. ACM,* vol. 28, pp. 22–23, Jan. 1985.

[19] C. L. Seitz *et al.*, "The architecture and programming of the Ametek Series 2010 Multicomputer," in *Proc. Third Conf. Hypercube Concurrent Comput. Appl.*, ACM, Jan. 1988, pp. 33–37.

[20] C. L. Seitz *et al.*, "Submicron systems architecture project semiannual technical report," Caltech Computer Science Tech. Rep., Caltech-CS-TR-88-18, p. 2 and pp. 11–12, Nov. 1988.

[21] C.-L. Wu and T. Feng, "On a class of multistage interconnection networks," *IEEE Trans. Comput.*, vol. C-29, pp. 694–702, Aug. 1980.



William J. Dally (S'78–M'86) received the B.S. degree in electrical engineering from Virginia Polytechnic Institute, the M.S. degree in electrical engineering from Stanford University, and the Ph.D. degree in computer science from Caltech.

He has worked at Bell Telephone Laboratories where he contributed to design of the BELLMAC-32 microprocessor. Later as a consultant to Bell Laboratories he helped design the MARS hardware accelerator. He was a Research Assistant and then a Research Fellow at Caltech where he designed the MOSSIM Simulation Engine and the Torus Routing Chip. He is currently an Associate Professor of Computer Science at the Massachusetts Institute of Technology where he directs a research group that is building the J-Machine, a fine-grain concurrent computer. His research interests include concurrent computing, computer architecture, computer-aided design, and VLSI design.