Design of Efficient and Easily Routable Generalized Connectors

Ching-Yi Lee and A. Yavuz Oruç, *Senior Member, IEEE*

Abstract — **This paper presents a new generalized connector with a very simple self-routing scheme. Unlike some of the recently reported generalized connectors, this generalized connector does not rely on an explicit use of a copy network; rather it replicates packets as it routes them through its stages to their destinations. In addition to its attractive routing scheme, this generalized connector can be constructed with $O(n \lg^2 n)$ bit-level constant fanin logic gates, $O(\lg^2 n)$ bit-level depth, and can realize any multicast assignment in $O(\lg^3 n)$ bit-level time[1].**

I. INTRODUCTION

Switching networks that realize multicast assignments are commonly referred to as *generalized connectors*, or *broadcast networks* and have been extensively studied in the literature [5,3,6,7,8]. The previously reported generalized connectors, were obtained by one of three methods. A survey of these methods is given in [2]. The method, which will be used in this paper, is based on recursive decompositions of generalized connectors. The rationale behind this approach is to generate copies of packets while they are being routed to their destinations, rather than use a separate copy network. This approach was introduced by Nassimi and Sahni [4] who constructed an n-input generalized connector with $O(kn^{1+1/k} \lg n)$ 2×2 crossbar switches and $O(k \lg n)$ depth for any $k, 1 \leq k \leq \lg n$. While the design of this generalized connector is based on elementary switches, its routing relies on a parallel computer model consisting of $n^{1+1/k}$ processors which are interconnected by a cube or perfect shuffle topology. The time to route multicast assignments on this model is $O(k \lg n)$ if it is assumed that a processor can perform certain arithmetic, logic and shift operations in $O(1)$ time. In gate-level, each processor would need $O(\lg n)$ logic gates, and experience $O(\lg n)$ gate delay[2]. Hence, the routing hardware for this generalized connection network exacts $O(n^{1+1/k} \lg n)$ logic gates, and it takes $O(k \lg^2 n)$ gate delays. We should also note that the realization of multicast assignments under this routing scheme requires sorting, and this means that incomplete assignments, i.e., those that do not involve all inputs can be realized only if the unused inputs are assigned to some dummy outputs.

In this paper, we give a self-routing generalized connector which is constructed in terms of simple logic gates. Unlike the routing scheme given in [4], routing on this generalized connector does not require sorting, and it proceeds from the inputs toward the outputs. For n inputs, this generalized connector uses $O(n \lg^2 n)$ logic gates, it has $O(\lg^2 n)$ gate delay, and can realize any multicast assignment in $O(\lg^3 n)$ time where the unit of time is a gate delay.

We note that all routing schemes described in this paper assume fixed-length packets and bit-synchronous operation. We also note that the generalized connection network presented in this paper does not handle multicast patterns in which an output may be connected to more than one input. On the other hand, such patterns can be decomposed into a sequence of multicast patterns in which no output is connected to more than one input. However, this approach may complicate our network constructions, in particular their self-routing property may no longer hold.

II. BASIC DEFINITIONS

An (n, q)-network is a directed acyclic graph with n distinguished source vertices, called inputs, q distinguished sink vertices, called outputs, and some internal vertices, called switching elements. If $n = q$, an (n, q)-network is abbreviated as an n-network. An *assignment* for a network is a pairing of its inputs with its outputs such that each output appears in at most one pair. An assignment consisting of k pairs is called a k-assignment. An assignment is called one-to-one or *unicast* if each input appears in at most one pair, and is called one-to-many, or *multicast*, otherwise. A network is said to *realize* an assignment if, for each pair (a, b) in the assignment, a path can be formed from input a to output b by setting the switching elements in the network with the constraint that the paths for no two pairs (a, b) and (c, d) overlap unless $a = c$. An (n, q)-network, $q \leq n$, is called an (n, q)-concentrator if, for each $k, 1 \leq k \leq q$, it can realize a unicast k-assignment between any k of its n inputs and the first k of its q outputs. An n-network is called a generalized n-connector if it can realize all multicast assignments between its inputs and its outputs.

Paths in a network will be established by specifying some routing information at its inputs. Each input holds its own destination information unless otherwise stated. The routing information for each input is accompanied by some binary coded message that is to be routed from that input to those output(s) specified in the routing information. A message and routing

Paper approved by G. P. O'Reilly, the Editor for Communications Switching of the IEEE Communications Society. Manuscript received February 28, 1992 and revised January 14, 1993. This work was supported in part by the Minta Martin Fund of the School of Engineering at the University of Maryland. This paper was presented in part at the 1992 International Conference on Parallel Processing, St. Charles, IL, August 1992.

The authors are with the Electrical Engineering Department, University of Maryland, College Park, MD 20742-3025.

[1] All logarithms are in base 2 unless otherwise stated, and $\lg n$ denotes the logarithm of n in base 2.

[2] The delay can be reduced to $O(\lg \lg n)$ but this requires a prefix circuit.

IEEE Log Number 9410097.

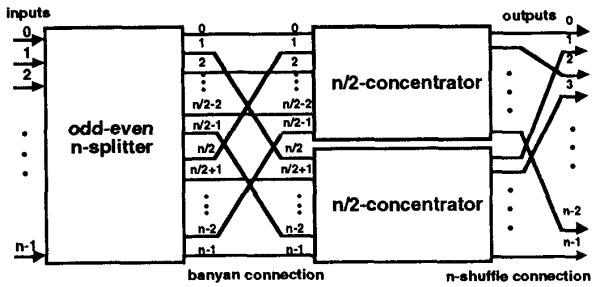

Fig. 1: The recursive construction of an n-concentrator.

information combined together will be termed a *packet*. For n inputs, the number of bits in the routing information will vary between 1 and $2n-2$ for all the networks described in this paper.

The performance of a network will be measured in terms of its hardware cost, depth and routing time. The *cost* of a network is the total number of constant fanin logic gates used in the network. The *depth* of a network is the largest number of constant fanin logic gates along a path from an input to an output. The *routing time* of a network is the maximum amount of time needed to set all the switching elements in the network for an assignment between the network's inputs and outputs, where time is measured in terms of constant fanin gate delays.

III. CONCENTRATOR CONSTRUCTION

In this section, we describe an n-concentrator with $O(n \lg n)$ constant fanin, logic gates, $O(\lg n)$ depth, and $O(\lg^2 n)$ routing time, all in bit level, where $n = 2^m$ for some positive integer m. In routing packets through this concentrator, it is assumed that there is a leading bit, called the *routing bit* r_i, for each packet at input i, $0 \leq i \leq n-1$. If $r_i = 1$, there is a packet to be concentrated at input i, i.e., a packet to be moved to the output side of the concentrator.

An n-concentrator is recursively constructed as shown in Fig. 1 by using an n-network, called an *n-splitter*. An n-network is an n-splitter if, for each $k, 1 \leq k \leq n$, it can realize a unicast k-assignment that pairs some $\lceil k/2 \rceil$ of any k of its n inputs with some $\lceil k/2 \rceil$ outputs in a fixed $n/2$-subset of outputs, and the remaining $\lfloor k/2 \rfloor$ inputs with some $\lfloor k/2 \rfloor$ outputs in the other $n/2$-subset of outputs[3]. An n-splitter is called an odd-even n-splitter if the two fixed $n/2$-subsets of outputs comprise odd numbered outputs and even numbered outputs, respectively. With this definition of an n-splitter, it can easily be shown that the network in Fig. 1 is an n-concentrator (See [2]).

To construct an odd-even n-splitter, we use an array of $n/2$ 2×2 switches and an $(n, n/2)$-network, called an $(n, n/2)$-balancer, put in parallel. Fig. 2 shows this odd-even n-splitter

[3] $\lceil x \rceil$ denotes the smallest integer greater than or equal to x, and $\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x.

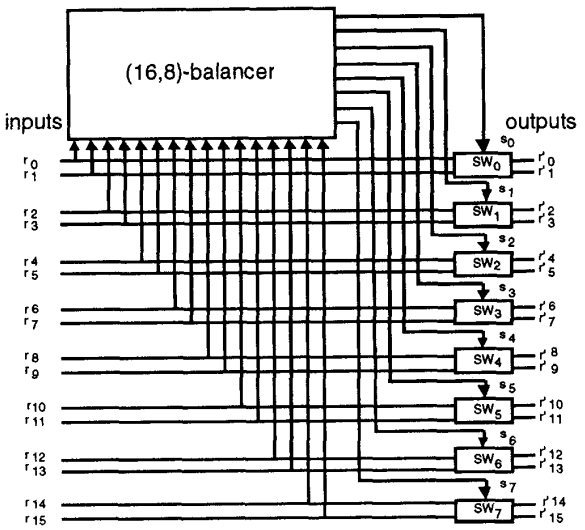

Fig. 2: An odd-even n-splitter for $n = 16$.

for $n = 16$. The inputs $2i$ and $2i+1$ of the odd-even n-splitter are connected to the inputs of switch SW_i, and the setting of SW_i is controlled by the binary input s_i, $0 \leq i \leq n/2-1$. If s_i is 0, the inputs of SW_i are connected to its outputs straight through; otherwise, the inputs are crossed over and connected to the outputs. The balancer, whose inputs are the routing bits r_i's of the incoming packets decides the values of the control inputs to the $n/2$ switches. Let r'_i be the leading bit of the packet at output i after the setting of SW_i, $0 \leq i \leq n$. If $s_i = 0$, then $r'_{2i} = r_{2i}$ and $r'_{2i+1} = r_{2i+1}$, and if $s_i = 1$, then $r'_{2i} = r_{2i+1}$ and $r'_{2i+1} = r_{2i}$, $0 \leq i \leq n/2-1$.

An $(n, n/2)$-balancer receives n routing bits, $r_0, r_1, \ldots, r_{n-1}$, from which it determines the values of its $n/2$ outputs, $s_0, s_1, \ldots, s_{n/2-1}$. We use a $\lg n$-level binary tree to implement an $(n, n/2)$-balancer, which is operated by a routing scheme similar to that described in [1]. Fig. 3 shows this balancer for $n = 16$. A more detailed description of the balancer can be found in [2].

IV. THE GENERALIZED CONNECTOR

The generalized n-connector is constructed recursively as shown in Fig. 4, where the non-recursive part consists of a distributor stage and two $(n, n/2)$-concentrators. The distributor is an array of n $(1,2)$-copiers where each (1,2)-copier can map its input to none, one or both of its two outputs, and the concentrators can be realized by the construction given in the previous section. That this n-network is a generalized n-connector can easily be shown (See [2]).

Unlike the previously reported generalized connectors, this generalized n-connector does not use any dedicated copy network. Rather, it generates the copies of the packets as they are being routed to their destinations. As we shall see subse-

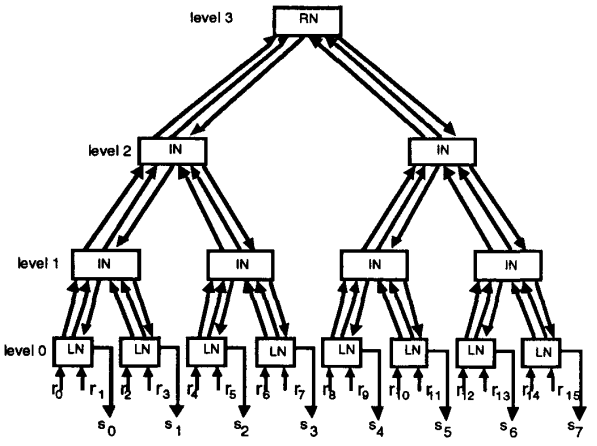

Fig. 3: The balancer with 16 inputs and 8 outputs.

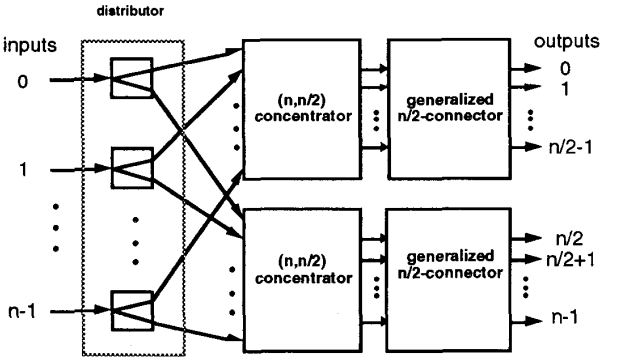

Fig. 4: The recursive construction of a generalized n-connector.

quently, this approach reduces the amount of destination address information that each input must use in order to route its packet to the outputs it desires.

For a given multicast assignment, a *multicast pattern* for an input i (or for a packet at that input) of a generalized n-connector is a binary sequence $(a_0, a_1, \ldots, a_{n-1})$, where $a_j = 1$ if and only if input i is paired with output j.

Proposition 1: If an input of a generalized n-connector specifies the outputs to which it is assigned independently of other inputs, then it must use at least n bits.

Proof: An input may have any one of 2^n multicast patterns. If the input is to distinguish among these 2^n patterns without relying on the specifications of the other inputs, it obviously needs $\lg 2^n = n$ bits. ||

We note that the independence assumption used here is very much like assuming that each input must use at least $\lg n$ bits to specify the output to which it is assigned in an n-connector where each input can be paired with at most one output. We also note that if the independence assumption is relaxed, then it is possible that all $(n+1)^n$ assignments for a generalized n-connector can be coded only with $\lg(n+1)^n = O(n \lg n)$ bits. In fact, this coding scheme can be enforced if the assignments are specified from outputs to inputs, since each output can only be connected to at most one input, and thus $O(\lg n)$ bits of source address per each output will suffice. However, in the generalized connector construction given here, it is assumed that the inputs are the active ports and initiate the connection requests. In this case, we do not know of any coding scheme that works with $O(\lg n)$ destination bits per input, and therefore, we assume that the inputs independently specify their multicast patterns by using $O(n)$ bits per input. We will describe two destination coding schemes, both using $O(n)$ destination bits per each input.

First, we establish the following proposition which will be used in the two destination coding schemes. Let $(a_0, a_1, \ldots, a_{n-1})$ be a multicast pattern of a packet at an input of the generalized n-connector as shown in Fig. 4. It is obvious that the packet is routed to the first half of outputs if and only if the elements $a_0, a_1, \ldots, a_{n/2-1}$ are not all zero, or equivalently, $a_0 + a_1 + \cdots + a_{n/2-1} = 1$,[4] and, it is routed to the second half of outputs if and only if $a_{n/2} + a_{n/2+1} + \cdots + a_{n-1} = 1$. More generally, the following proposition holds.

Suppose that the outputs in stage k of the network of Fig. 4 are partitioned into 2^k groups of $n/2^k$ consecutive outputs from top to bottom, $1 \le k \le \lg n$, and let O_i^k denote the ith group of outputs, $0 \le i \le 2^k - 1$. Define a binary variable d_i^k on k and $i, 0 \le i \le 2^k - 1$, so that $d_i^k = 1$ if and only if at least one or more of the a_j's, $ni/2^k \le j < n(i+1)/2^k$ are 1.

Proposition 2: Suppose that a packet at an input of the generalized n-connector shown in Fig. 4 has a multicast pattern $(a_0, a_1, \ldots, a_{n-1})$. Let d_i^k and O_i^k be defined as above, $0 \le i \le 2^k - 1$, $1 \le k \le \lg n$. Then, the packet would be routed to one or more outputs in O_i^k if and only if $d_i^k = 1$. ||

Hence, to route a packet with a multicast pattern $(a_0, a_1, \ldots, a_{n-1})$ through the generalized n-connector, the $(1,2)$-copiers along the path(s) of the packet can use d_i^k's to decide their settings. When it is recursively decomposed, the generalized n-connector has $\lg n$ distributor stages. If these distributor stages are numbered 1, 2, ..., $\lg n$ from left to right, then stage k has 2^{k-1} distributors, numbered 0, 1, ..., $2^{k-1} - 1$ from top to bottom, $1 \le k \le \lg n$. Each $(1,2)$-copier in the ith distributor at stage k uses the pair (d_{2i}^k, d_{2i+1}^k) it receives to decide its own setting as described in the next subsection.

A: Routing With $2n - 2$ Destination Bits

In the first coding scheme, each packet at an input is given a $(2n-2)$-bit destination address

$$(d_0^1, d_0^1, d_0^2, d_1^2, d_2^2, d_3^2, \ldots, d_0^{\lg n}, d_1^{\lg n}, \ldots, d_{n-1}^{\lg n}).$$

These $2n-2$ destination bits are defined as in Proposition 2. Given this coding of the destination addresses, the routing of multicast assignments is straightforward. For each value of k, $d_i^k, 0 \le i \le 2^k - 1$ form a 2^k-bit destination address used by

[4] Here + denotes the binary OR operation.

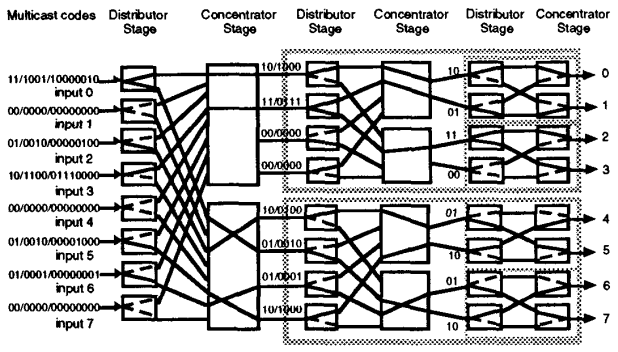

Fig. 5: A generalized 8-connector shown to realize a multicast assignment.

the distributors at stage k. If a $(1,2)$-copier in the ith distributor at stage k is on a path of a packet then it uses the two bits d^k_{2i} and d^k_{2i+1} it receives to route the packet, $0 \leq i \leq 2^{k-1}-1$, $1 \leq k \leq \lg n$. If the two bits are 01 then the packet is routed to the lower output of the copier, if it is 10 then it is routed to its upper output, and if it is 11 then it is routed to both outputs. Along with the packet, the copier also routes a valid destination code for the remaining stages on the path(s) of this packet. The determination of this destination code depends on the output(s) that the packet is routed to. If the packet is routed to the upper output of the copier, then the first half of the destination code for each of the remaining stages is retained, and the second half is discarded, and if it is routed to the lower output, then the second half of the destination code for each of the remaining stages is retained and the first half is discarded.

This routing scheme is illustrated in Fig. 5. The generalized 8-connector is obtained by decomposing the two generalized 4-connectors based on the construction given in Fig. 4.

B: Routing With n Destination Bits

The length of the destination code can be reduced from $2n-2$ to n, by increasing the complexity of routing at each copier node. In this case, the packet at each input carries its own n-bit multicast pattern $(a_0, a_1, \ldots, a_{n-1})$ as its destination address. The $(1,2)$-copiers must rely on more bits in order to determine which way to route the packets they receive. More specifically, the copiers in the kth distributor stage must examine $n/2^{k-1}$ bits, $1 \leq k \leq \lg n$, to determine their settings. Other than this, this multicast pattern coding scheme is very similar to the previous one, and its details are omitted.

V. PERFORMANCE ANALYSIS

We continue to assume in this section that all our networks have a power of two number of inputs and outputs. First, we compute the cost, depth, and routing time, $C_{conc}(n), D_{conc}(n), T_{conc}(n)$ for the n-concentrator. This n-concentrator is recursively constructed by using an odd-even n-splitter from which we have

$$\begin{aligned} C_{conc}(n) &= C_{split}(n) + 2C_{conc}(n/2), \\ D_{conc}(n) &= D_{split}(n) + D_{conc}(n/2), \\ T_{conc}(n) &= T_{split}(n) + T_{conc}(n/2), \end{aligned}$$

where $C_{split}(n), D_{split}(n), T_{split}(n)$ denote the cost, depth, and routing time of the odd-even n-splitter in that order. The odd-even n-splitter is composed of an array of $n/2$ 2×2 switches and an $(n, n/2)$-balancer, where the balancer is used to set the switches. Once the switches are set, the packets can pass through the switches. Hence, the routing time of the odd-even n-splitter is decided by the routing time of the balancer, and the depth of the splitter is the depth of a 2×2 switch. Assuming that a 2×2 switch can be implemented by using six bit-level logic gates with fanin 2, and depth 2, we have $C_{split}(n) = 3n + C_{balancer}(n), D_{split}(n) = 2, T_{split}(n) = T_{balancer}(n)$, where $C_{balancer}(n), T_{balancer}(n)$ denote the cost and routing time of the balancer, respectively. The $(n, n/2)$-balancer has $n-1$ switching elements and the longest path from an input to an output is $2 \lg n$. It can be verified that each of these switching elements can be implemented by no more than fourteen bit-level logic gates with fanin 2, and within any node, the largest number of 2-input logic gates along a path from an input to an output is 4. Hence, $C_{split}(n) \leq 17n$ and $T_{split}(n) \leq 8 \lg n$. Substituting all these expressions into the cost, depth and routing time recurrences for the n-concentrator, and solving the recurrences with $C_{conc}(2) = 6, D_{conc}(2) = 2$, and $T_{conc}(2) = 1$, we find $C_{conc}(n) \leq 17n \lg n - 14n, D_{conc}(n) \leq 2 \lg n, T_{conc}(n) \leq 4 \lg^2 n + 4 \lg n$.

Let $C_{gconn}(n), D_{gconn}(n)$, and $T_{gconn}(n)$ denote the cost, depth and routing time of the generalized n-connector in Fig. 4. By its construction, we have

$$\begin{aligned} C_{gconn}(n) &= C_{distr}(n) + 2C_{conc}(n) + 2C_{gconn}(n/2), \\ D_{gconn}(n) &= D_{distr}(n) + D_{conc}(n) + D_{gconn}(n/2), \\ T_{gconn}(n) &= T_{distr}(n) + T_{conc}(n) + T_{gconn}(n/2). \end{aligned}$$

The $(n, 2n)$-distributor is an array of n $(1,2)$-copiers and each $(1,2)$-copier can be implemented by two logic gates of fanin 2. Hence, $C_{distr}(n) = 2n$, $D_{distr}(n) = 1$ and $T_{distr}(n) = 1$. Substituting these, and the expressions for $C_{conc}(n), D_{conc}(n)$, and $T_{conc}(n)$ in the above recurrences, and solving them with $C_{gconn}(2) = D_{gconn}(2) = T_{gconn}(2) = 2$, we have

$$\begin{aligned} C_{gconn}(n) &\leq 17n \lg^2 n, \\ D_{gconn}(n) &= \lg^2 n + 2 \lg n + 1, \\ T_{gconn}(n) &\leq 4/3 \lg^3 n + 4 \lg^2 n + 11/3 \lg n. \end{aligned}$$

VI. CONCLUDING REMARKS

In this paper, we presented a generalized connection network. Unlike the previously reported generalized connectors, this network has a very simple design, and it can be routed very fast. It has $O(n \lg^2 n)$ cost, $O(\lg^2 n)$ depth, and $O(\lg^3 n)$ routing time, all in bit level. The paper also gave a construction for an n-concentrator that has $O(n \lg n)$ cost, $O(\lg n)$ depth, and $O(\lg^2 n)$ routing time, all in bit level. If these complexities can be reduced then the complexities of the generalized n-connector can also be scaled down by the same factor of reduction.

REFERENCES

[1] B. Douglass and A. Y. Oruç. Self-routing and route balancing in connection networks. UMIACS-TR-90-32,CS-TR-2421, Institute for Advanced Computer Studies, University of Maryland, College Park, MD., 1990.

[2] C.-Y. Lee and A. Y. Oruç. Design of efficient and easily routable generalized connectors. UMIACS-TR-92-22,CS-TR-2846, Institute for Advanced Computer Studies, University of Maryland, College Park, MD, 1992.

[3] G. M. Masson and B. W. Jordan. Generalized multistage connection networks. *Networks*, pp. 191–209, 1972.

[4] D. Nassimi and S. Sahni. Parallel permutation and sorting algorithms and a new generalized connection network. *Journal of the ACM*, pp. 642–667, July 1982.

[5] J. P. Ofman. A universal automaton. *Trans. of the Moscow Mathematical Society*, pp. 200–215, 1965.

[6] C. D. Thompson. Generalized connection networks for parallel processor intercommunication. *IEEE Trans. on Computers*, pp. 1119–1125, Dec. 1978.

[7] J. Turner. Design of a broadcast packet switching network. *IEEE Transactions on Communications*, pp. 734–743, June 1988.

[8] Y. Yang and Masson. Nonblocking broadcast switching networks. *IEEE Transactions on Computers*, pp. 1005–1015, Sept. 1991.

Ching-Yi Lee received the B.S. and M.S. degrees in Electrical Engineering from the National Taiwan University, Taiwan, in 1984 and 1988, respectively, and the Ph.D. degree in Electrical Engineering from University at Maryland at College Park, MD in 1992.

In 1993 he joined Telectronic International Inc., Rockville, MD as a systems engineer. His current research interests are in local area networks and wide area networks, including broadband ISDNs, ATM networks, and parallel and distributed processing.

A. Yavuz Oruç received the B.Sc. degree in Electrical Engineering from the Middle East Technical University, Ankara, Turkey in 1976, the M.Sc, degree in Electronics from the University of Wales, Cardiff, United Kingdom in 1978, and the Ph.D degree from Syracuse University, Syracuse, New York in 1983.

Since January 1988, he has been an associate professor in the Department of Electrical Engineering at the University of Maryland, College Park, Maryland. Prior to joining the University of Maryland, he was on the faculty of the Department of Electrical, Computer and Systems Engineering at Rensselaer Polytechnic Institute, Troy, New York.

His research interests include parallel computer and communication systems.

Dr. Oruç is a member of the IEEE Communications, Computer, and Information Theory Societies.