Report Number: WUCS-87-4

1987-04-01

System Testing of a Broadcast Packet Switch

Authors: Shabbir Khakoo and Jonathan S. Turner

Feng and Wu have described a fault diagnosis method for a class of multi-stage interconnection networks including the banyan, delta and omega networks. We extend their method to switch fabrics containing several cascaded networks. We also show the method can be applied to system in which the processor performing the testing does not have direct access to all input and output ports.

SYSTEM TESTING OF A BROADCAST PACKET SWITCH

Shabbir Khakoo and Jonathan S. Turner

WUCS-87-4

April, 1987

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Abstract

Feng and Wu have described a fault diagnosis method for a class of multi-stage interconnection networks including the banyan, delta and omega networks. We extend their method to switch fabrics containing several cascaded networks. We also show how the method can be applied to systems in which the processor performing the testing does not have direct access to all input and output ports.

This work supported by the National Science Foundation (grant DCI 8600947), Bell Communications Research, Italtel SIT and NEC.

SYSTEM TESTING OF A BROADCAST PACKET SWITCH

Shabbir Khakoo
Jonathan S. Turner

1. Introduction

In [1], Turner describes a packet switched communication system, which supports multipoint connection in addition to conventional point-to-point connections. The principal component of this system is an interconnection network called the *switch fabric*, which replicates and distributes broadcast packets, as well as routing point-to-point packets. The switch fabric comprises three cascaded binary routing networks. The nodes contain internal buffering and include control mechanisms for performing packet replication in addition to routing.

This paper is concerned with fault diagnosis of the switch fabric. It generalizes a method developed by Feng and Wu [2]. Their method was designed for unbuffered networks in which the testing processor has direct access to all input and output ports. We show how it can be extended to handle cascaded networks in which the testing processor has direct access to only one input/output port pair. For a switch fabric with n input and output ports and three cascaded networks, our method can detect and locate any single fault with no more than $6n$ test packets. Feng and Wu's method, when applied to a single network requires $2n$ test packets.

Section 2 of the paper gives a brief overview of the system described in [1] and describes our fault model. Section 3 briefly reviews Feng and Wu's method for diagnosing a single network and shows that it is capable of diagnosing all single faults using $2n$ tests under our fault model. Section 4 shows how Feng and Wu's method can be extended to switch fabrics containing several cascaded networks. We show that $(m+1)n$ tests are sufficient for testing m cascaded networks. Section 5 shows how our testing method can be applied when access by the testing processor is limited to one port pair. This requires an additional set of $2n$ access tests.

2. System Description

The overall structure of the system described in [1] is shown in Figure 1. Each Switch Module (SM) terminates up to 63 Fiber Optic Links (FOL). Data is carried on the FOLs in the form of large fixed-length packets. The *Packet Processors* (PP) perform link level protocol functions, including the determination of how each packet is routed. The *Switching Fabric* (SF) is a parallel packet switching interconnection network that provides routing of point-to-point packets plus replication and distribution of multipoint packets. The *Connection Processor* (CP), is responsible for establishing connections, including both point-to-point and multipoint connections. It is also responsible for testing the switch fabric.

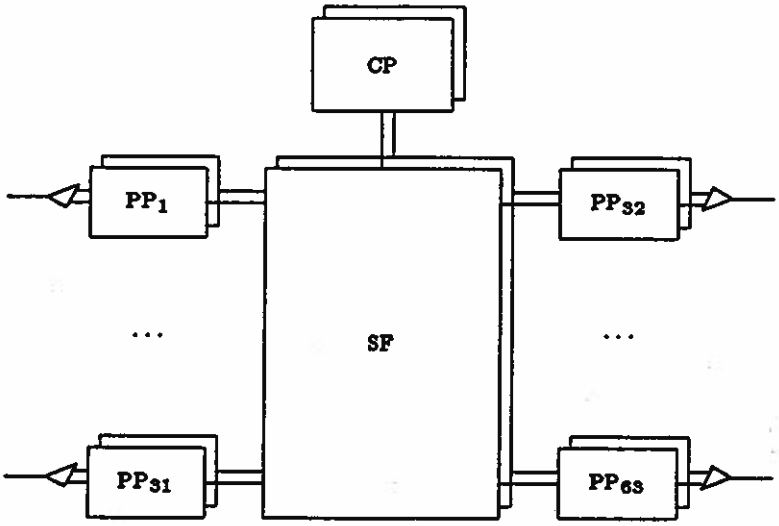

Figure 1: Switch Module

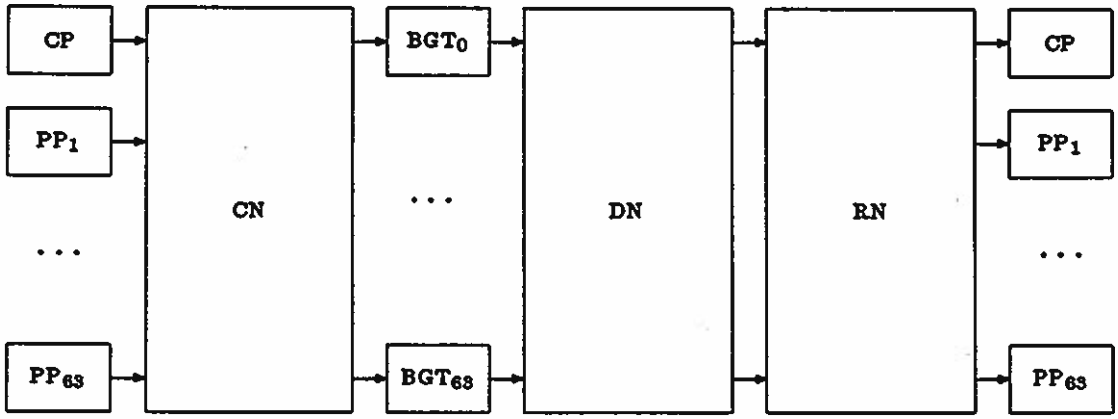

Figure 2: Switch Fabric

When a packet enters the SM, it is reformated by the addition of several new fields, the Routing field being the only one that concerns us here. The Routing field (RF) contains information needed to process a packet within the switch fabric. In the case of point-to-point packets, it includes an outgoing link number which is used to route the packet through the switch fabric. In the case of multipoint packets, it includes a *Fanout* field which specifies the number of outgoing links that must receive copies of the packet.

A block diagram of the Switch Fabric (SF) is given in Figure 2. It contains four major components, a *Copy Network*, a set of *Broadcast and Group Translators*, a *Distribution Network* and a *Routing Network*. The RN is a conventional binary routing network that routes packets based on an address field in the packet header. The DN has the same structure as the RN but instead of routing

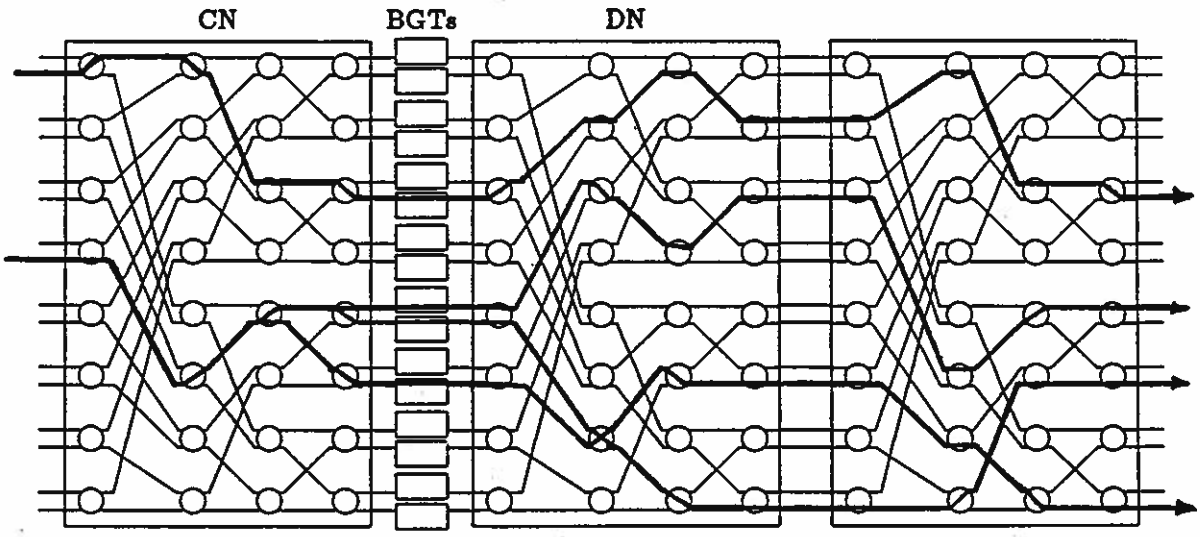

Figure 3: Example of Switch Fabric Operation

packets based on address information, it distributes them randomly across its outputs to prevent long term congestion that could otherwise occur in the RN. The CN performs the packet replication function required for multipoint connections. When a multipoint packet having k destinations enters the CN, it is replicated so that k copies of that packet are produced. Point-to-point packets pass through the CN without change. The Broadcast and Group Translators modify the headers of multipoint packets after replication, so that each copy is routed to a distinct outgoing PP.

Figure 3 illustrates the operation of the switch fabric (to simplify the figure, we have shown a switch fabric with just sixteen input and output ports). Two packets are shown entering the CN. The top one is a point-to-point packet with a destination address of 5. It follows an arbitrary path through the CN and DN before being routed to output port 5 by the RN. The nodes in the RN route the packet based on successive bits of the destination address. The upper output port (port 0) is selected at the first node in the RN because the first bit of the destination address is 0. The lower output port (port 1) is selected at the second node because the second bit of the destination address is 1, and so forth. The second packet entering the CN is a multipoint packet with a fanout of 3. The CN produces the required number of copies, then the BTCs assign destination addresses of 7, 9 and 13. The packets pass through the DN taking arbitrary paths and then are routed by the RN to the specified outgoing PPs. For a more detailed description of the system, see [1].

Testing of the switch fabric is performed by the Connection Processor (CP). The only direct access that the CP has to the switch fabric is through the port pair by which it's connected. To perform a complete test, it therefore requires the cooperation of the Packet Processors (PP). A test packet is sent by the CP to a specified PP (through the switch fabric). Upon receipt, the PP takes a field from the body of the packet, puts it in the routing field and sends the packet back to the switch fabric. It is received by a second PP, which puts the CP's address in the routing field and sends it once more to the switch fabric, which delivers it to the CP. The middle leg of this journey is the main part of the test. The first and last legs are just a mechanism that gives the CP effective access to all the ports.

The nodes in the switch fabric are moderately complex devices, containing buffering for several packets and a substantial amount of control circuitry. This seems to imply that one must consider

a large class of failure modes in testing them. We side-step this issue by assuming that each node contains sufficient internal redundancy that failures can be detected and cause the node to simply shut down. This can be achieved by replicating control circuitry and using coding techniques on the data path. Since the bulk of the circuitry in the nodes is expected to reside in the buffering, this need not be excessively expensive.

3. Testing a Single Network with Full Access

Feng and Wu have developed a method for testing a single network in which the testing processor has direct access to all input and output ports. In this section, we briefly review their method, adapting it to our fault model and using it to introduce the notations we will use.

We start by defining the structure of a network more precisely. Let k be some positive integer and $n = 2^k$. The networks we are interested in can be modeled as a directed acyclic graph $G = (V, E)$ where

$$V = \{s(j), t(j) \mid 0 \leq j \leq n-1\} \cup \{u_i(j) \mid 0 \leq i \leq k-1,\ 0 \leq j \leq (n/2)-1\}$$

The vertices $s(j)$ are the *source vertices* and correspond to the input ports of the network. The vertices $t(j)$ are the *sink vertices* and correspond to the output ports of the network. The vertices $u_i(j)$ are the internal vertices that correspond to the nodes of the network. Each of the vertices $u_i(j)$ has two incoming edges and two outgoing edges. We specify edges in a slightly unconventional way. Each edge is defined as an ordered pair of *ports* where ports are associated with vertices. Every source vertex $s(j)$ has a single port $\alpha_k(j)$. Every sink vertex $t(j)$ has a single port $\beta_0(j)$. Every internal vertex $u_i(j)$ has four ports, $\alpha_i(2j), \alpha_i(2j+1), \beta_{i+1}(2j), \beta_{i+1}(2j+1)$. With these definitions, we can define the edge set as follows,

$$\begin{aligned} E \;=\; & \{(\alpha_k(j), \beta_k(j)), (\alpha_0(j), \beta_0(j)) \mid 0 \leq j \leq n-1\} \cup \\ & \{(\alpha_i(j), \beta_i(\sigma_i(j))) \mid 1 \leq i \leq k-1,\ 0 \leq j \leq n-1\} \end{aligned}$$

where σ is a successor function. Our results can be applied to networks defined by a wide range of different successor functions, but for definiteness, we focus on a particular one. If $b_{k-1}, \ldots, b_0$ is the binary representation of j, then $\rho_i(j)$ is defined as the integer whose binary representation is $b_{k-1} \ldots, b_i, b_0, b_{i-1}, \ldots, b_1$. We can view this as a right rotation of the low order i bits of j. We let $\sigma_i(j) = \rho_{i+1}(j)$. Note, that with our definitions any edge can be identified by giving just one of its endpoints. We will often use this as a convenient shorthand.

We can illustrate these definitions by referring to Figure 3. The packet shown at the top of the figure passes through vertices $u_4(0), u_3(0), u_2(2)$ and $u_1(2)$ of the Copy Network and travels across edges $\alpha_4(1), \alpha_3(0), \alpha_2(1), \alpha_1(4)$ and $\alpha_0(5)$.

For our purposes, the key property satisfied these networks is that there is exactly one directed path connecting any source-sink pair. This in turn implies, that there is at most one path between any pair of nodes and at most one path between any pair of edges.

We need a few more definitions in order to define test sets. First, we define an iterated version of $\sigma_i(j)$; let $\sigma_i^0(j) = j$ and $\sigma_i^h(j) = \sigma_{i-1}^{h-1}(\sigma_i(j))$ for $1 \leq h \leq i$. Next, if $b_{k-1}, \ldots, b_0$ is the binary representation of j we let $\overline{\rho}_i(j)$ be the integer whose binary representation is $b_{k-1} \ldots, b_i, \overline{b}_0, b_{i-1}, \ldots, b_1$ and we let $\overline{\sigma}_i(j) = \overline{\rho}_{i+1}(j)$. Finally, we let $\overline{\sigma}_i^0(j) = j$ and $\overline{\sigma}_i^h(j) = \overline{\sigma}_{i-1}^{h-1}(\overline{\sigma}_i(j))$ for $1 \leq h \leq i$.

A *test* for a network is a pair of integers (τ_0, τ_1), which specifies the transmission of a test packet along the unique path from $s(\tau_0)$ to $t(\tau_1)$. A test $\tau = (\tau_0, \tau_1)$ fails if and only if there is a faulty link or node on the path from $s(\tau_0)$ to $t(\tau_1)$. A *test set* is just a collection of tests.

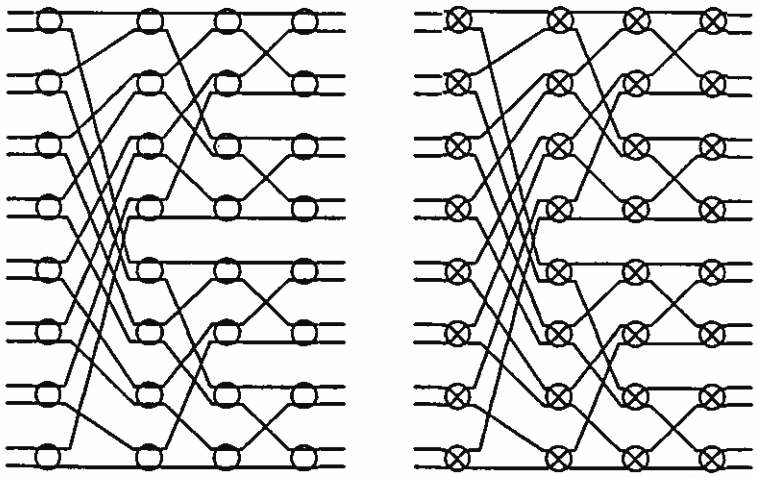

Figure 4: Two Test Phases for Single Network

Consider the test $\tau = (\tau_0, \tau_1)$ where $\tau_1 = \sigma_k^k(\tau_0)$. The path specified by this test satisfies the following property: for $0 \leq i \leq k-1$ if $\beta_{i+1}(j)$ is on the path then $\alpha_i(j)$ is also on the path. Informally, if the path enters a node through its 'top' (bottom) port it also exits through the top (bottom) port. Feng and Wu specify a test set consisting of two phases. Phase 0 contains n tests of the form $(\tau, \sigma_k^k(\tau))$ for $0 \leq \tau \leq n-1$. Phase 1 contains n tests of the form $(\tau, \overline{\sigma}_k^k(\tau))$ for $0 \leq \tau \leq n-1$. This is illustrated in Figure 4. Note that within each phase, the paths partition the edge set of the network.

Under our fault model, a single link failure will cause the failure of a single test in each phase. Let $(\tau_0, \sigma_k^k(\tau_0))$ be the failing test in phase 0 and $(\tau_1, \overline{\sigma}_k^k(\tau_1))$ be the failing test in phase 1. There is exactly one edge that is common to the paths specified by these two tests. (If there were two common edges, then there would have to be two paths in the network between those two edges.) To find the faulty link, we find the value of i for which $\sigma_k^i(\tau_0) = \overline{\sigma}_k^i(\tau_1)$ and let $j = \sigma_k^i(\tau_0)$. The failing link is then $\alpha_{k-i-1}(j)$.

A node failure causes the failure of two tests in each phase. There is exactly one node common to the paths specified by the failing tests in each phase. (If there were two nodes, there would have to be two paths joining those two nodes.) If $(\tau_0, \sigma_k^k(\tau_0))$ and $(\tau_1, \sigma_k^k(\tau_1))$ are the two failing test packets from the first phase, we can identify the faulty node, by finding the value of i for which $\lfloor \sigma_k^i(\tau_0)/2 \rfloor = \lfloor \sigma_k^i(\tau_1)/2 \rfloor$ and let $j = \lfloor \sigma_k^i(\tau_0)/2 \rfloor$. The failing node is then $u_{k-i-1}(j)$.

We comment that this testing method is easily extended to self-routing networks with base b nodes for $b > 2$. There are again two test phases. The first set routes packets straight through each node. The second phase simply rotates the paths by one port position at each node. This is sufficient to isolate any single link or single node failures.

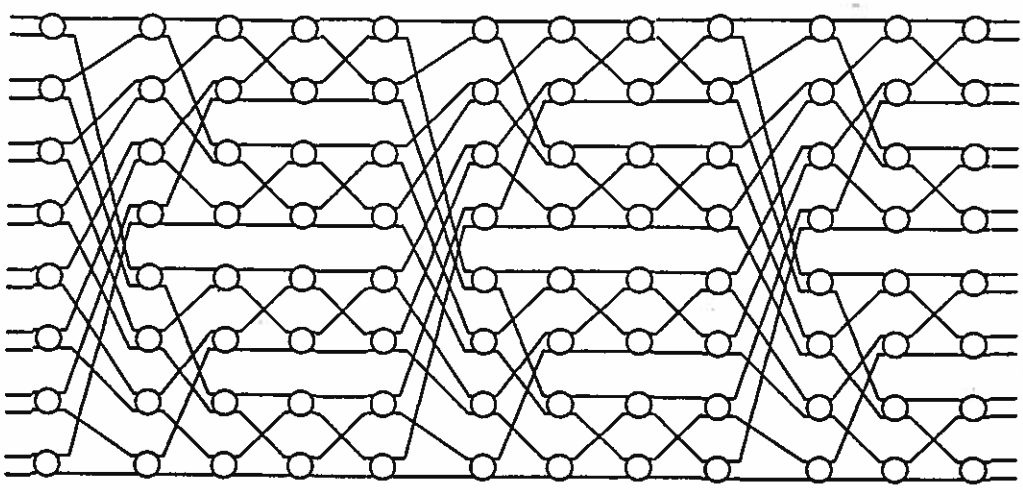

Figure 5: Three Cascade

4. Testing Cascaded Networks

An *m-cascade* is a sequence of m networks, $(G_1, \ldots, G_m)$ where each $G_r = (V_r, E_r)$ has vertex set

$$V_r = \{s_r(j), t_r(j) \mid 0 \le j \le n-1\} \cup \{u_{r,i}(j) \mid 0 \le i \le k-1,\ 0 \le j \le (n/2)-1\}$$

Every source vertex $s_r(j)$ has a port $\alpha_{r,k}(j)$. Every sink vertex $t_r(j)$ has a port $\beta_{r,0}(j)$. Every internal vertex $u_{r,i}(j)$ has four ports, $\alpha_{r,i}(2j), \alpha_{r,i}(2j+1), \beta_{r,i+1}(2j), \beta_{r,i+1}(2j+1)$. The edge set is

$$\begin{aligned} E_r \;=\; & \{(\alpha_{r,k}(j), \beta_{r,k}(j)), (\alpha_{r,0}(j), \beta_{r,0}(j) \mid 0 \le j \le n-1\} \cup \\ & \{(\alpha_{r,i}(j), \beta_{r,i}(\sigma_i(j))) \mid 0 \le i \le k-1,\ 0 \le j \le n-1\} \end{aligned}$$

where σ is the successor function defined earlier. We connect networks together by identifying the sinks in network r with the sources in network $r+1$. In particular, we define $s_{r+1}(j) = t_r(\chi(j))$ for $1 \le r \le m-1$. The function χ is called the connection function and specifies the mapping used between networks. The obvious approach of using the identity function makes it impossible to distinguish faults in the last stage of one network from those in the first stage of the succeeding network. Consequently, we use $\chi(j) = \rho_2(j)$. These definitions are illustrated in Figure 5 which shows a 3-cascade.

A single test for an m cascade is an $m+1$-tuple $\tau = (\tau_0, \ldots, \tau_{m+1})$. This specifies a test packet that follows the unique path that starts at $s_1(\chi(\tau_0))$ and passes through $t_r(\tau_r)$ for $1 \le r \le m$. Our test set for an m cascade has $m+1$ phases. For $0 \le r \le m$, phase r consists of n tests of the form $\tau^r = (\tau^r_0, \ldots, \tau^r_{m+1})$ where $0 \le \tau^r_0 \le n-1$ and

$$\tau^r_q = \begin{cases} \sigma^k_k(\chi(\tau^r_{q-1})) & \text{for } 1 \le q \le m \text{ and } q \ne r \\ \overline{\sigma}^k_k(\chi(\tau^r_{q-1})) & \text{for } q = r \end{cases}$$

This is illustrated in Figure 6. Note that the tests within each phase are edge disjoint.

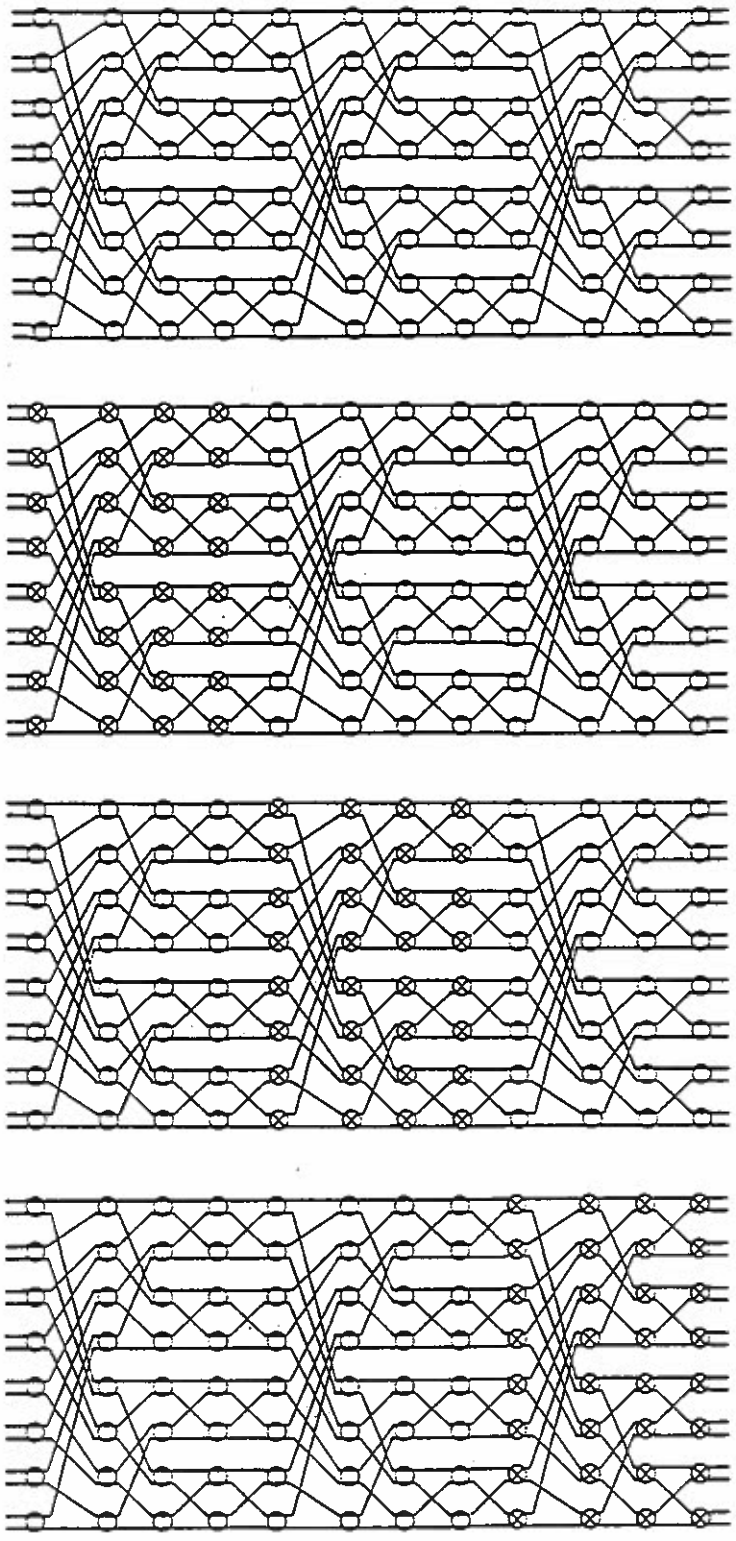

Figure 6: Four Test Phases for Cascaded Networks

A failure of a link causes exactly one test in each phase to fail. There is exactly one link that is common to all these paths. (If there were two common links then there would be two paths within one of the networks that were joined by two paths.) So we need only find that common link to identify the fault. A failure of a node causes two tests to fail in each phase. Again, there is only one node common to the paths used by all these tests and we need only find that node to identify the fault. An example illustrating the identification of a faulty link is given in Figure 7

5. Testing Networks With Limited Access

As mentioned in section 2, we have an immediate interest in a system in which the testing processor has limited access to the ports of the switch fabric. Effective access to all the ports is provided by Packet Processors (PP) ability to relay test packets to and from the Connection Processor (CP). There remains the question of how to verify the paths between the CP and PPs.

We assume the CP is connected to source 0 and sink 0 of an m-cascade with n pairs of ports. For the purposes of this section, a test is a $(2m+1)$-tuple $(\tau_0, \ldots, \tau_{2m})$. This specifies a test packet using the path from starting at $s_0(\tau_0)$ passing through $t_r(\tau_r)$ for $1 \leq r \leq m$, then passing through the network a second time starting from $s_m(\tau_m)$ and passing through $t_r(\tau_r)$ for $m+1 \leq r \leq 2m$.

Our objective in these *access tests* described below is not to verify all paths from the CP to the PPs, but merely to verify that there is some usable path from the CP to each PP. If we succeed in doing this, we proceed to the test of the full m-cascade as described in the previous section, using the verified access paths to get to and from the PPs. If we cannot identify a usable path, we attempt to localize the fault as far as possible. It turns out that it is not possible to isolate the fault to a single node or link. The best that can be done in general is to localize the fault to two links and two nodes.

We verify access to the PPs by a two phase test. The first phase contains n tests of the form $(\tau_0, \ldots, \tau_{2m})$ where $\tau_m = j$ for $0 \leq j \leq n-1$ and $\tau_r = 0$ for $r \neq m$. The second phase contains n tests of the form $(\tau_0, \ldots, \tau_{2m})$ where $\tau_m = j$ for $0 \leq j \leq n-1$, $\tau_0 = \tau_{2m} = 0$ and $\tau_r = n-1$ for $r \notin \{0, m, 2m\}$. We'll refer to the paths used by the first phase tests as the *high paths* and the paths used by the second phase tests as the *low paths*. Figure 8 gives an example of the high and low paths to and from a particular PP.

Note that the links connecting the CP to the switch fabric are used in every test path as are two nodes adjacent to the CP. If these links or nodes fail then all the access tests fail. Since these are the only links or nodes common to all tests, they are the only ones implicated in the case that all tests fail. Next consider some PP, say PP_j. The links connecting it to the switch fabric are used in both the high and low paths to it, as are the two nodes adjacent to it. These are the only nodes and links common to both of these paths, with the exception of the nodes and links adjacent to the CP. From these observations, we see that a failure of any internal link or node (that is a link or node that is not adjacent to either the CP or a PP) is insufficient to prevent access to any PP. So, if the only failures are internal, the full test set discussed in the previous section can be employed for fault identification and isolation, using the verified access paths to relay the test packets to and from the PPs.

We've already noted that the failure of any of the links or nodes adjacent to the CP can be isolated to that set of links and nodes, but there is no way of isolating the fault any further than this (indeed, the fault may be in the CP). The failure of a link adjacent to one of the PPs causes

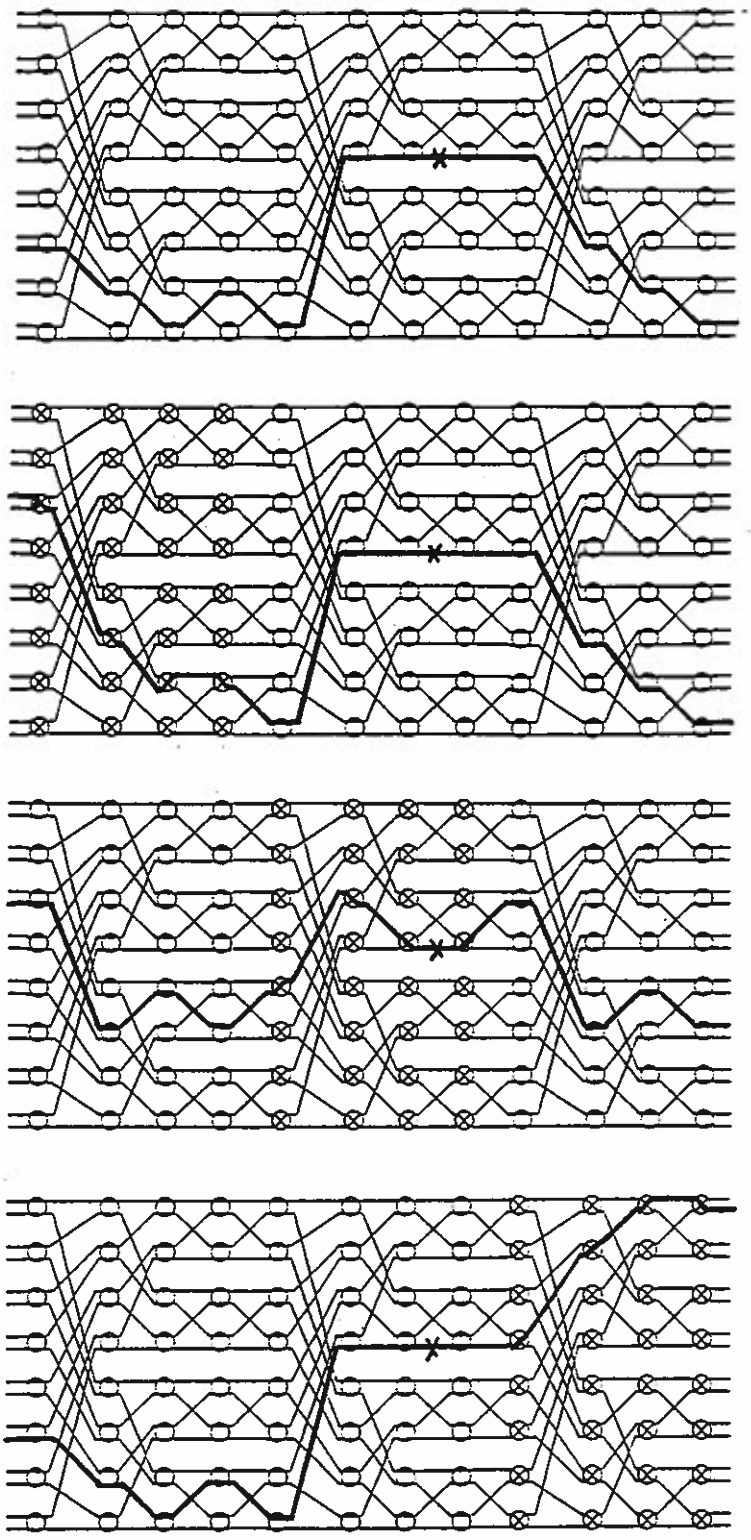

Figure 7: Example Showing Identification of Faulty Link

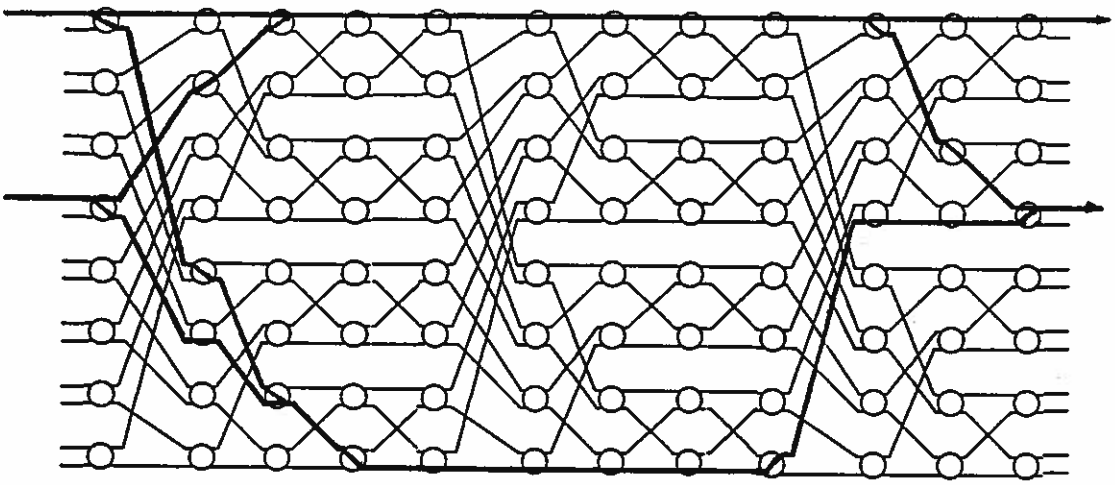

Figure 8: High and Low Paths

both paths to that PP to fail, but if the nodes adjacent to the PP are not faulty, the access tests to the other PP connected to those nodes will succeed. Hence, we can isolate this fault to one of the two links connecting the PP to the switch fabric. If an external node fails, then both access paths to both adjacent PPs will fail, permitting us to isolate the node failure to one of the two adjacent nodes. We note that these faults can be isolated to a single node by changing the mapping between PPs and input ports so that two PPs that are adjacent to a common node on the input side of the switch fabric are not also adjacent to a common node on the output side. A simple way to do this is to connect PP_j to input port $\chi(j)$.

6. Summary

In this paper, we have extended the testing method of Feng and Wu [2] to cascaded networks with limited port access. We have shown that for such networks, $(m+3)n$ tests are sufficient for access verification and fault detection/location of single link and node faults in m-cascades.

References

[1] J. S. Turner, "Design of a Broadcast Packet Switching Network," Washington University, Computer Science Department, WUCS-85-4, 3/85.

[2] Tse-yun Feng and Chuan-lin Wu, "Fault-Diagnosis for a Class of Multistage Interconnection Networks," *EEE Transactions on Computers*, vol. c-30, no. 10, 10/83.

[3] Tse-yun Feng, "A Survey of Interconnection Networks," *Computer*, vol. 14, no. 12, 12/83, 12-30.

[4] D. P. Agrawal, "Testing and Fault Tolerance of Multistage Interconnection Networks," *Computer*, 4/82.