

Testing a K -ary N -cube Interconnection Network

Seelan Kumarasamy, Sandeep K. Gupta and Melvin A. Breuer

Department of Electrical Engineering-Systems

University of Southern California

Los Angeles, CA 90089-2560

{seelan@scf, sandeep@boole, mb@poisson}.usc.edu

Abstract

In this paper we present a methodology for testing k -ary, n -cube direct interconnection networks that support deterministic routing, wormhole switching and virtual channel flow control. The approach is based on functional testing of the individual wormhole routers in the network. Based on a canonical architecture for the wormhole router, a functional fault model is developed from a comprehensive set of functional faults that describe the major modes of failure of the router. Functional tests are developed to detect these faults by defining appropriate sets of communication messages. A modified version of these functional tests can be applied to the periphery of a network to test all the routers within the network. The ability of the functional tests to detect physical faults in a router circuit is validated by fault simulation and the fault coverage is high.

1 Introduction

High performance direct interconnection networks [1] are widely used in multiprocessor, multicomputer, distributed system architectures and in network switches and serve as the communication backbone in the design of scalable systems. Direct interconnection networks are constructed by connecting together network routers. Since routers are large circuits, traditional automated test generation techniques are inadequate in generating tests for networks. The problem of efficiently testing high performance direct interconnection networks has received little attention in the literature and no techniques for systematic and efficient on-line or off-line testing of these networks have been published. We propose an efficient functional testing methodology for deterministic wormhole routers that is based on a functional model of the router and show that the proposed tests can serve as the basis for testing the overall network. In the following sections basic interconnection network and functional testing concepts are reviewed. The functional model of the router, the fault model, fault classification, functional tests developed and an approach to testing a network using functional tests for an individual router

are described. Fault coverage results obtained via simulation of a test bench router implementation are given.

2 Background

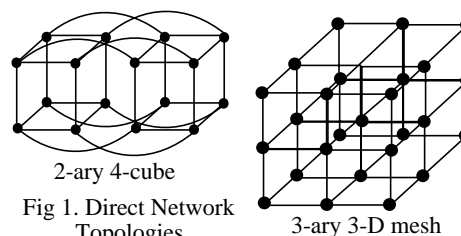


Fig 1. Direct Network Topologies

Direct interconnection networks are characterized by topology, flow control, routing and switching. The topology of a network defines how the nodes are interconnected by channels. Most popular direct network topologies can be categorized as either n -dimensional meshes or k -ary n -cubes. Figure 1 illustrates several direct network topologies. Communication between routers is performed by sending messages, which are broken into packets for transmission. Packets contain two or more flits - the smallest unit of information on which flow control is performed. Flow control deals with the allocation of channels and buffers to a packet as it travels through the network. Virtual channels provide multiple buffers for each channel in the network. A network using virtual channel flow control [2] organizes the buffers it associates with each channel into several lanes and buffers in each lane can be allocated independently. Routing determines the path selected by the packet through intermediate nodes to reach its destination in the network and the routing algorithm indicates the next channel to be used at each node. For deterministic routing the path is completely determined by the source and destination addresses. Switching moves data from an input channel and places it on an output of the router. For wormhole switching the header flit is blocked if it encounters a channel already in use and flow control within the network blocks the trailing flits which remain in buffers along the established route. In a network using wormhole switching and virtual channel flow control a blocked message holds only a single lane idle and can be passed using buffers of the

remaining lanes.

The functional testing approach used here is based on a functional model of the system. *Functional models* reflect the functional specifications of a system and are largely independent of implementation. *Functional testing* is used to validate the operation of a system against its functional specifications and can be approached with a specific functional fault model or without a fault model. *Functional fault models* attempt to represent effects of physical faults on the operation on a functionally modeled system and the faulty behavior induced should realistically match the faulty behavior induced by physical faults. Tests generated to detect faults for a good functional fault model should provide high coverage for stuck-line faults in a structural model of the system. Functional fault models have been used successfully in testing RAMs and programmable devices such as microprocessors [3].

3 Router Functional Model - Architecture of a Canonical Router

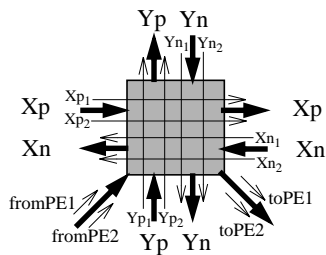


Fig 2. Router channels and lanes

Our functional level model of a wormhole router is based on a canonical architecture for a wormhole router described in [4]. See Figs. 2, 3. Wormhole routers perform switching, routing, flow control, multiplexing of physical channels and inter-router signaling. Essential components in the canonical router architecture are the flow control unit, address decoders, routing arbitration logic, virtual channel controllers and the crossbar. The flow control unit performs flow control between routers, stopping a message in place in the network if necessary, and computes the updated header. Signaling flow control information across a channel incurs delay, so sufficient buffering to capture the data in flight on must be available. The unit consists of an external flow controller (XFC) and an internal flow controller (IFC). The XFC controls the pipelined inter-node data flow and provides buffering required to prevent data loss when data-flow is stopped and resumed. The IFC controls intranode flow control (in conjunction with the virtual channel controller) and is responsible for updating the routing information in the relative addressing scheme used. Control logic in the IFC also serves the function of tail detection. The

address decoder (AD) serves to detect an incoming packet header flit, decode the routing address information and generate requests for a crossbar connection to the appropriate output. The AD consists of a *header detect and decode* sub-block and a *request generation* sub-block. The routing decision block (RD) chooses the path and connects and disconnects the input to/from an appropriate output based on the routing algorithm and channel status information. The RD performs request arbitration and output lane allocation and sets up corresponding crossbar (CB) connections to satisfy packet routing requirements. It consists of a request arbitrator, lane allocator and connection, input and output lane status and acknowledgment generation sub-blocks. The *request arbitrator* arbitrates among requests from address decoders for the available router outputs. Successful requests proceed to the *lane allocator* which assigns output lanes to requesting input lanes from the request arbitrator. The *connection* sub-block latches granted input lane - output lane connections and drives the crossbar. The *i/o lane status* sub-block maintains availability state information for each input/output lane while the *acknowledgment generator* generates acknowledgments for the address decoder for granted connection requests.

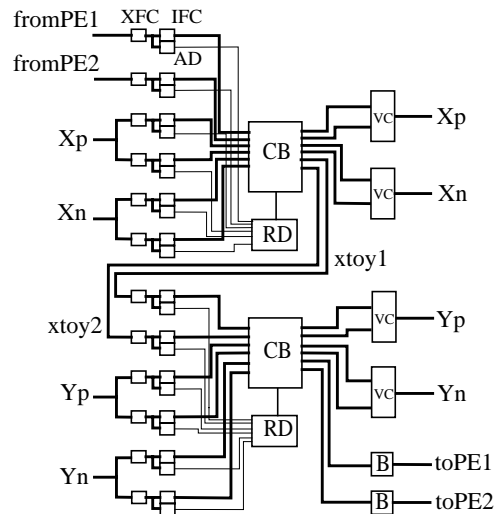


Fig 3. Canonical Architecture for a Wormhole Router

The virtual channel controller (VC) multiplexes the physical link amongst a set of virtual lanes and controls intranode data flow in cooperation with the IFCs of the lanes, providing a set of virtual channels that can progress independently. The VC operates in a collection phase and a delivery phase. In the *collection* phase the VC collects flits from the appropriate IFCs via the CB, sends one flit to the output buffer directly and stores the rest in secondary buffers in the VC. In the next cycle the *delivery* phase begins and the data from the output buffer is transmitted while a flit moves from a VC

buffer to the output buffer. This is repeated in subsequent cycles until all the flits have been transmitted. The VC then returns to the collection phase and the process is repeated.

4 Functional Fault Model, Classification and Tests

A comprehensive fault model has been developed by analysis of the router's basic functional blocks using the lowest functional sub-block level to determine possible failure modes of the sub-blocks. The complete functional fault model is described in [5]. The model, consisting of over 35 functional faults, is organized according the functional block and sub-block in which each fault appears. A functional description of each fault, possible causes of the fault originating from low-level (stuck line and bridging) faults with reference to a gate level implementation of the router and the fault effect is presented. The faults identified include faults in the multiplexor input selection in the VC; faults in the header, tail and zero address detection, buffer status as well as address update in the IFC and AD; faults in arbitration for crossbar connections and connection allocation and faults associated with granting connection requests in the RDB. Faults on flow control handshake signals between functional blocks and between routers are also included. It is assumed that the fault model and functional tests are developed for a router architecture that supports 2 virtual lanes per channel.

Functional tests have been derived to detect each functional fault in all possible instances of the corresponding functional block in the canonical architecture. We have classified most of the functional faults defined in the model into 5 main classes based on *equivalence and similarity of test requirement*. The functional tests for each fault class and the additional handshake, incorrect arbitration and incorrect allocation faults in the model are described next. It is assumed that at most one functional fault may be present in the router at any given time. The notation and the functional tests described pertain to the x -dimension of the router. Functional tests applied to a router are classified as either non-blocking tests or blocking tests. *Non-blocking* tests are applied to an input channel with the input channel in the free state with respect to the flit destination output while *blocking* tests are applied to the input channel with the input channel in the blocked state with respect to the flit destination output, i.e. the flit destination output lane status is busy. The input channel is subsequently freed. To test for faults in *Class 1* and *Class 2* and the *output lane stuck-at-busy* (olsb) handshake fault in lane k of input channel i , i_k , where

$k \in \{1, 2\}$ and $i \in \{fromPE1, fromPE2, Xp, Xn\}$ we need to send packets having all possible values of header flits from i_k . These are non-blocking functional tests. If all header flits sent are received at the expected outputs with the correct data then i_k is free of Class 1 and Class 2 faults and olsb faults. To test for the *output lane stuck-at-free* (olsf) handshake fault in lane k of input channel i , i_k , header, data and tail flits are sent from i_k while channel i is blocked (blocking test) with respect to the flit destination output. Channel i is then freed. If flits are received at expected outputs with correct data when the lane is freed and no flit is received at the output while the lane is blocked then input channel i_k is free of the olsf handshake fault. The functional test to detect faults in *Class 3* and *Class 4* and the olsb fault in the RDB involve sending packets (headers) from input virtual lanes to output lanes, making and breaking the possible crossbar connections. To test input channel-output combination i_k to j_l where $k, l \in \{1, 2\}$, $i \in \{fromPE1, fromPE2, Xp, Xn\}$ and $j \in \{Xp, Xn, xtoy1, xtoy2\}$, if $l=1$, we need to send a packet from i_k , with appropriate header values to be received at output destination j . If $l=2$ then we need to send a header from an input lane other than k in channel i , to output channel j , (lane $l=1$) before sending a header from i_k to output channel j lane $l=2$. These functional tests are non-blocking tests. If, for all such i, j, k and l combinations, the header flit is received at the expected output lane and is correctly updated then the x -dimension of the router is free of Class 3 and Class 4 faults and olsb faults. To test for the olsf handshake fault flits (headers) must be sent in on input virtual lanes to output lanes as for Class 3 and 4 and olsb faults, but with the input channel blocked (*blocking test*) with respect to the output destination of the flit. To test input channel-output combination i_k to j_l , if $l=1$, we need to send a packet from input i_k , with appropriate header values to be received at output destination j , while i is blocked with respect to output destination j . i_k is subsequently freed with respect to the flit output destination (j_l). If $l=2$ then we need to send a header from an input lane other than k in channel i , to output channel j , (lane $l=1$) before sending a header from i_k to output channel j lane $l=2$, while i_k is blocked with respect to the output destination of the flit in $i_k(j_2)$. i_k is subsequently freed with respect to j_2 .

The functional tests for detecting *incorrect arbitration* faults and *incorrect allocation* faults in the RDB for a particular output destination lane involve sending packets (header flits) from several input channels to be received at the output destination while the input channels are blocked with respect to that

output. To test for incorrect arbitration faults for output destination $j \in \{Xp, Xn, xtoy1, xtoy2\}$, lane $k, l \in \{1, 2\}$, j_k , and for incorrect allocation faults for output destination $j \in \{Xp, Xn, xtoy1, xtoy2\}$, we need to send headers (with different data) for output destination j from all sets of combinations of possible input channels $i \in \{fromPE1, fromPE2, Xp, Xn\}$ that may be connected to j , while the input channels are blocked with respect to destination output j . The channels are then freed with respect to lane k of output destination $j, j_k, k \in \{1, 2\}$ for the incorrect arbitration test, or simultaneously freed with respect to (both lanes of) destination output j for the incorrect allocation test. If the flit from the correct (highest priority) input channel is received first at the output when the channels are freed with respect to an output destination lane for all combinations of possible input channels and for all output destinations, then the x -dimension of the router is free of incorrect arbitration faults while if the flits from the input channels are received at the correct output destination lanes when the channels are simultaneously freed with respect to both output destination lanes, then the x -dimension of the router is free of incorrect allocation faults.

To test for *Class 5* faults and handshake faults in the VC for destination output $j \in \{Xp, Xn, xtoy1, xtoy2\}$ we need to send header and data flits from input channel $i \in \{Xp, Xn\}$, lane $k=1$ and $k=2$ with the appropriate header and data values (different for each lane) to be received at output destination j , while input channel i is blocked with respect to output destination j , then subsequently free input channel i with respect to (both lanes of) output destination j .

The functional tests described detect all the functional faults defined in the functional fault model. In Procedure 1 *Send_header* (x, y, i_k) latches a header vector with value (x, y) at input channel i , lane k . *Send_data* (d, i_k) latches a data vector with value d at input channel i , lane k . *Send_tail* (i_k) latches a tail vector (tail bit set) at input channel i lane k . *block* (i) blocks input virtual channel i with respect to both lanes in the destination output of the flit in i . *free* (i) frees input virtual channel i with respect to both lanes in the destination output of the flit in i .

5 Testing an Interconnection Network

In defining faults in the fault model we assume that when a functional test is applied to a faulty router the error appears on an output (channel) of the router. The

```
(a) For k = 1 to 2 do          /* non-blocking test */
    For j = 0 to 63 do
        Send_header (x = j, y = 0, ik)
        Send_tail (ik)
        Send_header (x = 0, y = any value, ik)
        Send_tail (ik)
        Send_header (x > 0, y = 0, ik)
        Send_data (allones, ik)
        Send_data (zero, ik)
        Send_tail (ik)

(b) For k = 1 to 2 do        /* blocking test */
    block (i)                /* w.r.t. output dest. Xp */
    Send_header (x > 0, y = 0, ik)
    free (i)
    block (i)
    Send_data (allones, ik)
    free (i)
    block (i)
    Send_data (allzeros, ik)
    free (i)
    block (i)
    Send_tail (ik)
    free (i)
    block (i)                /* w.r.t. output dest. xtoy */
    Send_header (x = 0, y = any value, ik)
    free (i)
    Send_tail (ik)
```

Procedure 1: Blocking and non-blocking tests for Class 1 & 2 and handshake faults in channel $i=Xp$.

error at an output has a composite functional value or *functional characteristic* of the form v/v_f which takes values "correct_flit/no flit or corrupted_flit" received. All functional tests described here can be categorized as *single flit functional tests*. Flits are applied to one or more input channels of a router while the corresponding input buffers are empty. These tests do not detect faults that propagate errors to the reverse handshake (channel status) signals between the XFC and IFC. Because the buffers in the XFC are empty, the *empty* channel status signal between routers is set and the XFC does not propagate further any errors propagated to the *empty* signal from the IFC. Since the tests are single flit tests and flits are never held in the XFC buffers, tests that propagate an error (from an IFC) on the *empty* channel status signal between the IFC and XFC will not propagate the error forward to an output channel. Hence faults that produce such errors are not detected.

We are developing a theoretical framework for applying functional tests to router within an interconnection network. For a dimension order routed mesh interconnection network, I , and a channel i_k , a test flit can be routed from a periphery node or other network primary input to i_k . Since input channels of a router in I are free with respect to all destination

outputs, single flit non-blocking tests can be applied to a set of input virtual channels (that do not share a physical channel) of the router. An error in a flit, when received on an input channel of a fault-free router in the network will propagate to the next channel in the network on the path of the test flit in a fault free network and eventually reach a network primary output. Hence functional faults detectable by single flit non-blocking tests to an individual router can be detected in a router embedded in a network by applying the single flit non-blocking tests to the router within the network. In a network that is free of functional faults detectable by single flit non-blocking tests it is possible to set up the network to block a channel i_k of a router with respect to any destination output. Hence single flit blocking tests can be applied to a set of input virtual channels of a router within a network if the routers in the network are free of functional faults detectable by single flit non-blocking tests. A functional error produced will propagate to a peripheral node output or primary output of the network. Thus, assuming a single functional fault in the network, any functional fault in an individual router that is detectable by single flit blocking or non-blocking tests can be detected in a router embedded in a network by applying the single flit blocking and non-blocking tests to the router in the network along with messages to set up the test conditions at the router being tested. The entire network can be tested by testing each router in the network in turn in this fashion.

6 Test Bench and Fault Coverage Simulation Results

Finally, the ability of the functional tests developed to detect physical faults in a router circuit is validated by fault simulation on a gate level simulation model of a dimension order wormhole router for a 2D mesh interconnection network supporting two virtual channels per physical channel, implemented based on the canonical architecture. The design is a modified version of the Planar Adaptive Router [6] with a

Table 1: Functional Test Fault Coverage Results

Functional Block	# Faults	% Fault Coverage
IFC & AD	878-1232	92.4
RD	3014	89.4
VC	816	93.6
XFC	4308	86.8
CB	3984	84.1

synchronous interface. The Mentor Graphics Quickfault tool was used to perform fault simulation. Fault coverage of the functional tests for the the router is validated against the stuck-line fault model. Table 1 gives the coverage results obtained for the different functional blocks. Fault coverage for the XFC and CB are given as well although no functional tests or fault model have been specifically developed for these blocks.

7 Conclusions

Systematic and efficient testing of high performance direct interconnection networks has received little attention in recent literature. We have presented an approach to testing direct interconnection networks based on functional testing of individual network routers. A comprehensive functional fault model has been developed and functional tests have been developed to detect functional faults in the model. Functional faults in individual routers in an interconnection network can be detected by a simple extension to functional tests to the individual routers applied from the inputs of the network. Each router in the interconnection network can be tested in turn and this approach can be used to test the entire network. The functional tests developed for this purpose have been validated against a physical fault based model by fault simulation and the fault coverage obtained is high. We are presently studying more efficient approaches to using the individual router functional tests to test a network.

8 References

1. J. Duato, S. Yalamanchili and L. Ni, *Interconnection Networks: An Engineering Approach*. IEEE Computer Society Press, 1997.
2. W. J. Dally, "Virtual Channel Flow Control," *IEEE Trans. Parallel Dist. Systems*, vol 3, no. 2, pp. 194-205, Mar. 1992.
3. D. Brahme and J. A. Abraham, "Functional Testing of Microprocessors," *IEEE Trans. Comp.*, vol C-33, no. 6, pp.475-485, June 1984.
4. A. A. Chien, "A Cost and Speed Model for k-ary n-cube Wormhole Routers," in *Proc. Symp. Hot Interconnects*, IEEE Computer Society, Aug. 1993.
5. S. Kumarasamy, S.K. Gupta and M.A. Breuer, "Testing a K-ary, N-cube Interconnection Network," Technical Report CENG 98-15, Dept. of Electrical Engineering, Univ. of Southern California, July 1998.
6. K. Aoyama, "Design issues in implementing an adaptive router," Master's thesis, U. Illinois, Dept. of Comp. Sc., 1304 W. Springfield Ave., Urbana, Illinois, Jan. 1993.