RESEARCH ARTICLE OPEN

Detecting Temporary and Permanent Faults in NOC Using FTDR Algorithm

Manjula G
PG Scholar, VLSI Design
Tagore Institute of Engineering and Technology, Attur
Tamil Nadu - India

ABSTRACT
Decreasing size of integrated circuit leads to increase in susceptibility to temporary and permanent faults. This project proposes a fault - tolerant solution for a buffer less network-on-chip, with on-line fault-diagnosis mechanism is used to detect both temporary and permanent faults. Hybrid automatic repeat request of forward error correction and link-level error control scheme are used to handle temporary faults and a reinforcement-learning based fault-tolerant deflection routing (FTDR) algorithm is used to tolerate permanent faults without deadlock and live lock. A hierarchical routing-table-based algorithm (FTDR-H) is also presented to reduce the area overhead of the FTDR. By backtracking method, Reconfigurable FTDR algorithm is used with to clear the seek structures and compute the new path with dummy node. Routing table is updated to analyze the data under real application workloads.BCH code is used to detect the multibit errors. Modelsim simulator is used to verify an existing and proposed method.
Keywords:-Temporary And Permanent Faults; Fault Tolerance; On-Line Fault Diagnosis; Backtrack Method; FTDR Algorithm.

I. INTRODUCTION

Network on Chip (NoC) has emerged as a promising solution for on-chip communication to enable integrating various processors and on-chip memories into a single chip. Network-on-chip is communication subsystem on an integrated circuit typically between IP cores in a system on a chip (SoC). Difficult timing closure becomes the main problem among many design issues which is caused by long global wire delay. Long global wire delay and scalable issues are solved using this order, a packet based communication network which is known as Network-on-Chip (NoC). As the CMOS technology scales down to the nanometer domain, smaller size and higher frequencies increase higher of occurrence of intermittent and temporary faults, besides manufacturing defects and wear out effects which lead to permanent faults are also impossible to avoid. Faults come across NoC architectures, namely temporary faults, intermediate faults and permanent faults. These methods used to deal with these faults are flow control based and fault tolerant routing methods. Fault-tolerant routing algorithm should ensure “0 lost packets” in whatever fault patterns as long as a path exists In order to have higher speed and lower cost than a wormhole or a virtual channel router or bufferless routers are incorporated on NoC. In each input port, to except one input register there is no buffer in the bufferless router. Due to the buffers, deflection routing is utilized in the buffer less router to route packets to neighboring routers immediately without buffering in the router. The complete feature of deflection routing provides the potential to route packets to avoid faulty links/routers and achieve fault-tolerance. Now, in order to avoid temporary and permanent faults a fault-tolerant solution with an on-line fault diagnosis mechanism, a link-level error control scheme and a fault tolerant routing algorithm are proposed for the buffer less NoC.

II. RELATED WORKS

A. Fault Detection Mechanisms for Network on Chip

In order to run a fault-tolerant system smoothly the first thing to be done is to detect the location of the faults. The fault detection mechanism should also be able to distinguish temporary faults from permanent faults. There are few works to focus on detecting temporary faults and permanent faults at the meantime. Three techniques are used to handle temporary faults in NoC and Forward error correction, Automatic repeat request, and Hybrid-ARQ are used to detect and correct those faults. Temporary faults are handled at both transport level and link-level error control. It is found to have errors in ARQ error control scheme if the packet is retransmitted. Packets are retransmitted until it is received without errors. Through a cyclic redundancy check (CRC), the error detection is implemented. Then before transmitting, a simple error detecting code is applied to the packet, at the receiver side, checksum is calculated to ensure that no error has occurred. If the checksum does not calculate the value, the packet is retransmitted. FEC is used to increase the performance of error control. Error-correcting codes

(ECCs) are used to achieve add redundancy to the packet, which allows an amount of bit-errors to be detected and corrected at the receiver side.

B. NoC Architecture and Fault Diagnosis

The NoC architecture is based on a 2-D mesh topology, Nostrum NoC. Each processing element is attached to a router (R), as shown in Fig. 1. The difference from the ordinary 2-D mesh is that the boundary output is connected to the input of the same router. This can be viewed as an additional packet buffer. All incoming packets are prioritized according to their hop counts, which record the number of hops the packet been routed. The router makes routing decision for each arriving packet from the highest priority to the lowest.

1. Packet format

The basic data transfer unit in this paper is a packet. The original packet format, which is compatible with a multicore NoC platform, is shown in Fig. 2(a). A packet, which has 114 bits, contains a 34-bit head and an 80-bit payload. A valid bit (V) is used to mark a packet valid or not. Relative addressing is used for the source and destination address fields (SA and DA) which have 12 bits (six bits for row/column address), respectively.

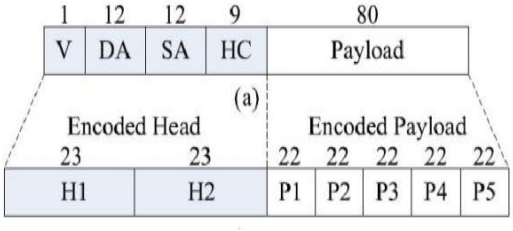

Fig 1: Encoded packet format.

The hop counter field (HC, nine bits) records the number of hops the packet has been routed. In order to detect and correct errors in transmission, SECDED Hamming codes are used to encode the head and payload, respectively. The encoded packet format, which has 156 bits, is shown in Fig. 2(b). The head is divided into two parts and Hamming (23, 17) code is used to encode each part. The payload is divided into five parts, each of which is encoded with Hamming (22, 16) code.

2. Link-Level Fault Detection And Protection

To perform fault diagnosis, single error correcting double error detecting hamming code (SECDED) technique is used. SECDED can correct single bit errors and detect double bit errors. The above mentioned flow chart can be explained as, if it is a single bit error whatever the error may be it will correct it automatically.

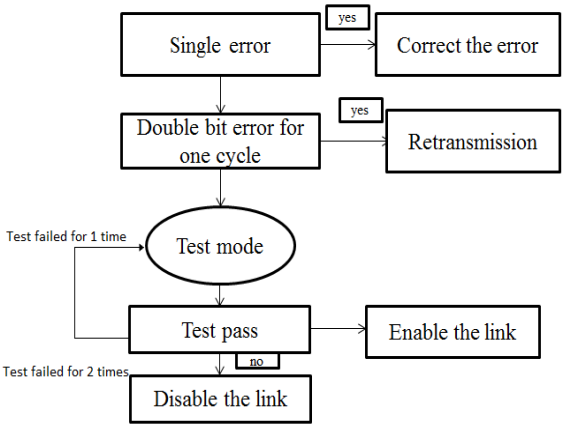

Fig 2: Fault diagnosis process.

If it detects a two-bit error in any one part of the encoding packet for one cycle, which is considered as a temporary fault, it will require the upstream router to retransmit the packet. If the syndromes of two consecutive received packets are the same, which means the retransmitted packet contains the same two-bit error, in order to check whether the link contains real permanent faults, the router enters into a test mode to test this faulty link by applying test vectors. The test process will be conducted at most twice. In the test mode, the link being tested is temporarily disabled. If the downstream router detects any one of the four test vectors still containing the same error, the test process will be conducted again. If the second test still fails, the link is marked as a permanent faulty link. If all tests pass, the link will be enabled again.

3. Link-Level Error Control Scheme

A simple error-detecting code is applied to the packet before transmitting, and at the receiver side a checksum will be calculated to ensure that no error has occurred. A hybrid ARQ scheme for error control is used to tolerate temporary faults during packets transmission. In the case of a single-bit error in any part of the packet, the error can be corrected after the packet has been decoded. If any part of the packet contains a two bit error for one cycle, the router which receives the packet will require the router, which sends the packet to retransmit the packet. The hardware structure of the hybrid ARQ is shown below. Here for each input port of the router an input buffer (IBi) with two entries instead of the original one and the boundary input port of the boundary router has an input buffer with three entries. Additionally, a retransmission

buffer is used to buffer the packet which may be retransmitted.

After decoding, the packet will be written to retransmission buffer.A2-to-1 Here for each input port of the router we have an input buffer (IBi) with two entries instead of the original one and the boundary input port of the boundary router has an input buffer with three entries. Additionally, a retransmission buffer is used to buffer the packet which may be retransmitted. After decoding, the packet will be written to retransmission buffer.

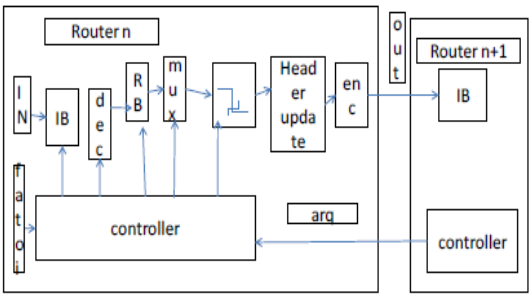

Fig 3: Hardware structure of link-level error control scheme

A 2-to-1 multiplexer is used to select to send a new packet or retransmit the last packet. A request signal arq is introduced between two neighboring routers to indicate whether the last packet should be retransmitted or not. The fault information transmission signal (fault to[i]) is used to disable the outgoing link i of the upstream router temporarily.

C. *FTDR Algorithm*

In order to distinguish temporary faults from permanent faults, the fault diagnosis process is shown in Fig. 3. The decoder will generate a syndrome, which contains the error information of the packet. If the decode detects a single-bit error in any one part of the encoding packet, it will correct it no matter which kind of faults it is. If it detects a two-bit error in any one part of the encoding packet for one cycle, which is considered as a temporary fault, it will require the upstream router to retransmit the packet. If the syndromes of two consecutive received packets are the same, which means the retransmitted packet contains the same two-bit error, in order to check whether the link contains real permanent faults, if all tests are passing, the link will be enabled again. In the test mode, the other links of the upstream and downstream routers can still transmit packets as normal. From this test process, temporary faults can be distinguished from permanent faults.

	North	East	South	West
R1	1	3	3	∞
R2	2	4	4	∞
R3	3	3	5	∞
R4	0	0	0	0
R5	3	1	3	∞
R6	4	2	4	∞
R7	3	3	1	∞
R8	4	2	4	∞
R9	5	3	5	∞

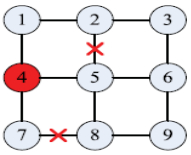

Fig 4: Reconfigured routing table update function

The routing table update function is shown in Algorithm

Fig 5: Update Function

Routing table update (dest_ ID,Q_ Value_ from, FON_from)
For dir £ {North, East, South, West}
If link (dir) is faulty then
For i=1 to N and I ≠ Switch_ ID
Table _entry (i)(dir) ←∞
If Neighbor (dir) has only one link not faulty then
For I =1 to N and (I≠ switch_ ID and I ≠ Neighbor_ID(dir) then
Table entry (i)(dir)←∞
If j £ {North, East, South, West}
If FON_ from (dir)(j)=1 then
for n £ {all switches along j through Neighbor(dir)}
Table_ entry(n)(dir)=table_ entry(n)(dir)+2;
If link(dir)from disable to enable then
For I =1 to N and I ≠ switch_ ID
Table_entry(i)(dir)←min_hops(switch_ID, i, dir)
IfQ_value_from(dir,dest_ID)≠0and dest_ID≠switch_ID then
Table_ entry(dest _ ID)(dir)←Q_ value_ from(dir, dest_ID)

1. If there is no fault in the network, the routing table cannot be updated. If one link of the router is broken or temporarily disabled during testing, all table entries corresponding to this direction are set to "∞" (steps 2–4). After a learning period, the table entries will converge to a fixed value which denotes the minimum number of hops from m each port to each destination. Additionally, we use the two-hop fault information to reduce the average hop counts. If a router detects that one of its neighborly along direction d has only one link not faulty based on the two-hop fault information, the table entries from d to all destinations except y are set to "∞".

The pseudo code of the algorithm is shown in Algorithm

For each input packet (from the highest priority to the lowest priority)
1: *dest_ID*←get_dest_ID(*dx*, *dy*)
2: *Hops_to_Dest*←table_lookup(*dest_ID*)
3: *If Hops_to_Dest = (0,0,0,0) then*
4: Route packet to local port
5: *else*
6:*Q_value_to*(*input_Dir*,*dest_ID*)←1+min(*Hops_to_Dest*) //send Q-value
7: {*d productive*} ← get_prefer_Dir(*Hops_to_Dest*)

8: *if* there are free ports in {*dproductive*} *then*
9: Choose a free productive port with the smallest stress value to route the pack
10**:** *else*//all productive ports are not free
11: Choose a free port with the smallest stress value to route the packet
2. There are at most four packets reaching a router at the same time. The router makes routing decision from the highest priority packet to the lowest. The router first calculates the destination ID of the packet and looks up the routing table to check if the packet has reached the destination (steps 1–4). If the packet has not reached the destination, the router looks up the productive direction(s) with the minimum number of hops
to destination from the routing table and then chooses a free productive port with the smallest stress value to route the packet (steps 7–9). If there is no free productive port, the router chooses a free port with the smallest stress value to route the packet, which can balance the network traffic loads (steps 10 and 11).

III. PROPOSED ROUTING WITH BACKTRACKING

At start up, the network is assumed fault free and packets are normally sent from the source PE to the target PE. Once the packet reaches its destination, the target PE must deliver an acknowledgment packet to the source PE. If the *ack* packet is not delivered in a given time, the source PE assumes the original path is faulty, and triggers the proposed path search method. This method, executed at runtime, comprises three steps, (*i*) seek new path; (*ii*) backtrack the new path; (*iii*) clear the seek structures and compute the new path.

In the first step of the method, ***seek step***, *if* the source PE identifies that it is not able to transfer packets to a target PE, this PE generates a *seek request*, asking for a fault-free path to the target PE.

The second step, ***backtrack step*** is executed only once, and starts when the target router is reached by the first received seek. All subsequent seeks are discarded. The target router injects *backtrack packet* into the NoC, where its payload correspond to the ports selected by each router to reach the target router. *Backtrack packet* uses source routing, following the reverse path followed by the seek step. Backtrack ends when the source router is reached and backtrack packet is sent to the source embedded processor to check the validity of the path.

The third step is named ***compute path and clear seek***. The path followed by backtrack packet might create a deadlock. For this reason, the source embedded processor checks the path, for instance [N N E EE S S W E], for invalid turns. When there is an invalid turn, it forces the packet to change the physical channel (or virtual channel), breaking the dependency cycle.

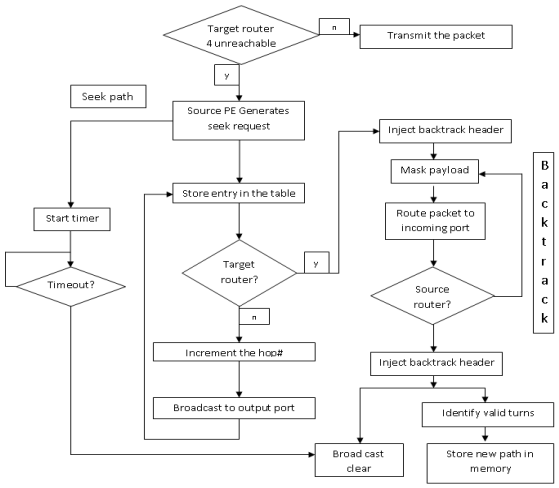

Fig 5: Flowchart of the routing method

Dummy nodes are used in faulty or faultless links. It is used to find the shortest path. It ensures "0 packet loss" at the case of faulty link. Total execution time of the path computation is small.

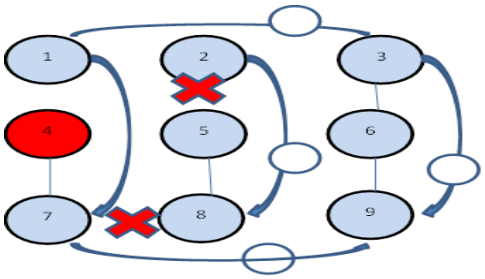

Fig 6: create new shortest path with dummy node

Algorithm

Back track(int k)

{

new path path (sp);

if (new path< best_path);

(best_path:=new_path);

(best_souln:=copy(sp);

}

back track (k+1)

A. Results

The experimental results of temporary and permanent faults and encoded packet type with dummy node through a reinforcement learning method as shown in the figs

.

a) Encoder and decoder packets with backtracking,
b) Router with dummy node.

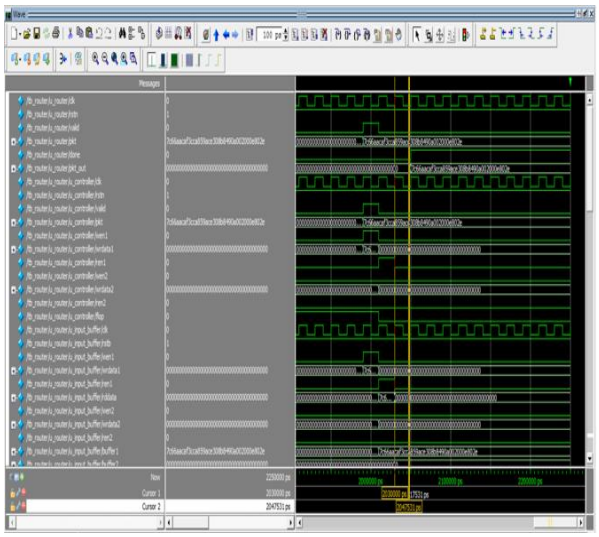

Fig a

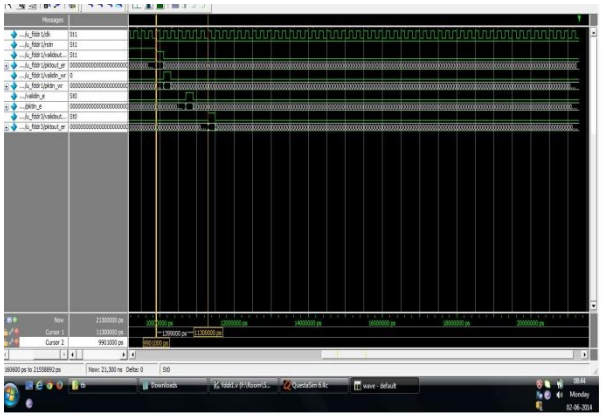

Fig b

IV. CONCLUSION

In this project, a fault-tolerant solution for a buffer less NoC to protect it from both temporary and permanent faults on the links has been explained.

a) An on-line fault diagnosis mechanism utilizes SECDED Hamming code to detect both temporary and permanent faults.

b) A hybrid ARQ/FEC scheme, which can achieve a high fault rate, is proposed to tolerate temporary errors during transmission.

c) The FTDR algorithm is proposed which guarantees "0 lost packets" reconfigures the routing table with dummy node through a reinforcement learning method to route packets avoiding permanent faults.

d) Backtrack routing method is used to create a new path and route packets through the shortest path.

REFERENCES

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip inter connection networks," in Proc. 38th Annu. Design Autom. Conf.,2001,pp.

[2] C. Constantinescu, "Trends and challenges in VLSI circuit reliability,"IEEE Micro, vol. 23, no. 4, pp. 14–19, Jul.–Aug. 2003.

[3] Y. H. Kang, T.-J. Kwon, and J. Draper, "Fault-tolerant flow controlin on-chip net works," in Proc. 4th ACM/IEEE Int. Netw.-Chip Symp.,May 2010.

[4] T. Theocharides, M. J. Irwin, L. Benini, and G. De Micheli, "Analysis of error recovery schemes fornet works on chips," IEEE Design Test Comput, vol. 22, no. 5, pp. 434–442, Sep.– Oct. 2005.

[5] S. Pasricha, Y. Zou, D. Connors, "OE+IOE: A novel turn model based fault tolerant routing scheme for networks-on chip," in Hardw./Softw. Code sign Syst. Synth., Oct. 2010,

[6] A. Patooghy and S. G. Miremadi, "XYX: A power & performance efficient fault-tolerant routing algorithm for net work on chip," inProc.17th Euromicro Int. Parallel, Distrib. Netw.-Based Process. Conf., 2009,pp.

[7]Z. Lu, M. Zhong, and A. Jantsch,"Evaluation of on-chip networks using deflection routing," in Proc. 16th ACM Great Lakes Symp. VLSI,2006, pp.

[8] T. Moscibroda and O. Mutlu, "A case for buffer less routing in onchipnetworks," in Proc. 36th Annu. Int. Symp. Comput, 2009.

[9] M. Hayenga, N. E. Jerger, and M. Lipasti, "SCARAB: A single cycle adaptive routing and buffer less net work," in Proc. 42nd Annu.IEEE/ACM Int. Symp. Microarch., Dec. 2009, pp. 244.

[10] C. Feng, Z. Lu, A. Jantsch, J. Li, and M. Zhang, "A reconfigurable fault tolerant deflection routing algorithm based on reinforcement learning for network-on-chip," in Proc. 3rd Int. Workshop Netw. ChipArch., 2010, pp.

[11] H. Zimmer and A. Jantsch, "A fault model notation and error control scheme for switch-to-switch buses in a network-on-chip," in Proc. 1st IEEE/ACM/IFIP Int. Conf. Hard/ Softw. Code sign Syst.Synth., Oct. 2003, pp. 188–193.

[12] C. Feng, Z. Lu, A. Jantsch, J. Li, and M. Zhang, "FoN: Fault-on-neighboraware routing algorithm for networks-on-chip," in Proc.23rd IEEE Int. SoC Conf., Sep. 2010, pp. 441–446.

[13] A. Kohler, G. Schley, and M. Radetzki, "Fault tolerant network on chip switching with graceful performance degradation," IEEETrans.Comput. Aided Design Integ.