ISSN 2348-375X

Unique Journal of Engineering and Advanced Sciences

Available online: www.ujconline.net

Research Article

DESIGN OF A SELF-ADAPTIVE ON-CHIP PERMUTATION NETWORK FOR MULTIPROCESSOR SYSTEM-ON-CHIP

Swapna P Jose[1*], Nithin SS[2]

[1]Dept. Of ECE,University of Calicut, Nehru College of Engg and Research centre,Pampady,Kerala,India
[2]Dept. Of ECE,University of Calicut, Nehru College of Engg and Research centre,Pampady,Kerala,India

Received: 04-03-2014; Revised: 02-04-2014; Accepted: 01-05-2014

*Corresponding Author: **Swapna P Jose**
Dept. Of ECE, University of Calicut, Nehru College of Engg and Research centre, Pampady,Kerala,India Email: swapnapj@gmail.com

ABSTRACT

This paper presents the silicon-proven design of a novel on-chip network to support guaranteed traffic permutation along with a self-contained adaptive system for detecting and bypassing permanent errors in multiprocessor system-on-chip applications . The proposed network employs a pipelined circuit-switching approach combined with a dynamic path-setup scheme under a multistage network topology. The dynamic path-setup scheme enables runtime path arrangement for arbitrary traffic permutations.The circuit-switching approach offers a guarantee of permuted data and its compact overhead enables the benefit of stacking multiple networks. The proposed system reroutes data on erroneous links to a set of spare wires without interrupting the data flow. To detect permanent errors at runtime, a novel in-line test (ILT) method using spare wires and a test pattern generator is proposed. In addition, an improved syndrome storing-based detection (SSD) method is presented and compared to the ILT method.

Keywords: Guaranteed throughput, multistage interconnection network, permutation network, pipelined circuit-switching, Adaptive systems, digital systems, fault tolerance, forward error correction, integrated circuit interconnections, network-on-chip.

INTRODUCTION

A trend of multiprocessor system-on-chip (MPSoC) design being interconnected with on-chip networks is currently emerging for applications of parallel processing, scientific computing, and so on. Permutation traffic, a traffic pattern in which each input sends traffic to exactly one output and each output receives traffic from exactly one input, is one of the important traffic classes exhibited from on-chip multiprocessing applications. Many of the MPSoC applications (e.g., Turbo/LDPC decoding) compute in real-time, therefore, guaranteeing throughput (i.e., data lossless, predictable latency,guaranteed bandwidth, and in-order delivery) is critical for such permutation traffics[1-7].

To support permutation traffic patterns, on-chip permutation networks using application-aware routings are needed to achieve better performance compared to the general-purpose networks. Such application-aware routings cannot efficiently handle the dynamic changes of a permutation pattern, which is exhibited in many of the application phases. The difficulty lies in the design effort to compute the routing to support the permutation changes in runtime, as well as to guarantee the permutated traffics[6-11].

Most on-chip networks employ a packet-switching mechanism to deal with the conflict of permuted data. Their implementations use first-input first-output (FIFO) queues for the conflicting data. Regarding the topology, regular direct topologies, such as mesh and torus, are intuitively feasible for physical layout in a 2-D chip. On the contrary, the high wiring irregularity and the large router radix of indirect topologies such as Benes or Butterfly pose a challenge for physical implementation. However, an arbitrary permutation pattern with its intensive load on individual source-destination pairs stresses the regular topologies and that may lead to throughput degradation. In fact, indirect multistage topologies are preferred for on-chip traffic-permutation intensive applications. Regarding the switching technique, packet switching requires an excessive amount of on-chip power and area for the queuing buffers(FIFOs) with pre-computed queuing depth at the switching nodes and/or network interfaces[12-18].

The number of faults in on-chip links is expected to increase as technology scales further into the nanoscale regime. While most faults are temporary, about 20% of all errors are caused by permanent or intermittent faults. Error control coding (ECC) techniques are commonly used to address reliability issues in on-chip interconnects, but these techniques generally

target transient errors rather than permanent errors. In order to maintain coding strength in the presence of permanent errors, spare wires can be used to replace permanently erroneous wires. The introduction of spare wires requires the following: 1) reconfiguration control and logic for bypassing erroneous wires and 2) a protocol for synchronizing information between receiver and transmitter.

This paper presents a novel silicon-proven design of an on-chip permutation network to support guaranteed throughput of permutated traffics under arbitrary permutation. Clos network is a multistage switching network. The advantage of such network is that connection between a large number of input and output ports can be made by using only small-sized switches. Here 3 stage Clos network is used. Unlike those conventional packet-switching approaches, proposed on-chip network employs a circuit-switching mechanism with a dynamic path-setup scheme under a multistage network topology. The dynamic path setup tackles the challenge of runtime path arrangement for conflict-free permuted data. The pre-configured data paths enable a throughput guarantee. By removing the excessive overhead of queuing buffers, a compact implementation is achieved and stacking multiple networks to support concurrent permutations in runtime is feasible.

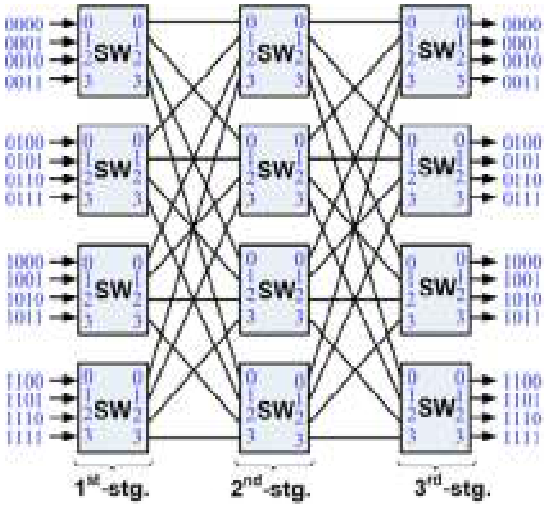

Figure 1: Proposed on-chip network topology with port addressing scheme.

This paper presents a system that uses spare wires to replace permanently erroneous wires without interrupting the data flow. To detect these permanent errors, propose a novel in-line test (ILT) method to test each adjacent pair of wires in a link for opens and shorts. These tests can be run periodically to ensure that each link's ECC capability is not being crippled by permanent errors. By testing every wire in the link, the ILT method also recovers resources from intermittent errors that were incorrectly flagged as permanent. In addition to the ILT method, here describe a number of important improvements to an alternative syndrome storing-based error detection (SSD) method, which is based on evaluation of consecutive code syndromes at the receiver. Syndromes are calculated during the decoding procedure and contain information on errors in the received words[19-23].

The rest of this paper is organized as follows. Section II presents the proposed on-chip network design with its dynamic path-setup scheme to support runtime path arrangement. Section III presents the proposed permanent-error detection methods in which the reconfigurable link framework which can be used as self-adaptive system for the proposed network is presented. In Section IV integration of the self-adaptive system in the network is discussed. Implementation details given in Section V. Section VI includes a discussion and finally, Section VII concludes this paper and outlines the further researches.

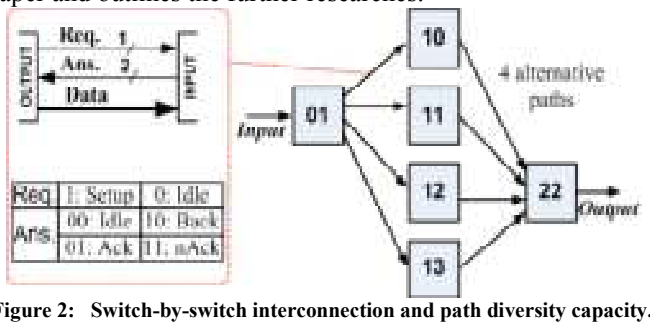

Figure 2: Switch-by-switch interconnection and path diversity capacity.

Proposed On-Chip Network Design

As motivated in Section I, the key idea of proposed on-chip network design is based on a pipelined circuit-switching approach with a dynamic path-setup scheme supporting runtime path arrangement. The network topology is first discussed. Then the designs of switching nodes are presented.

On-Chip Network Topology

Close network, a family of multistage networks, is applied to build scalable commercial multiprocessors with thousands of nodes in macrosystems. A typical three-stage Clos network is defined as C (n, m , p), where n represents the number of inputs in each of p first-stage switches and m is the number of second-stage switches. In order to support a parallelism degree of 16 as in most practical MPSoCs ,here proposed to use C (4, 4, 4) as a topology for the designed network(See Fig. 1). This network has a rearrangeable property that can realize all possible permutations between its input and outputs. The choice of the three-stage Close network with a modest number of middle-stage switches is to minimize implementation cost, whereas it still enables a rearrangeable property for the network[24-28].

A pipelined circuit-switching scheme is designed for use with the proposed network. This scheme has three phases: the setup, the transfer, and the release. A dynamic path-setup scheme supporting the runtime path arrangement occurs in the setup phase. In order to support this circuit-switching scheme, a switch-by-switch interconnection with its handshake signals is proposed, as shown in Fig. 2. The bit format of the handshake includes a 1-bit Request (Req) and a 2-bit Answer (Ans). Req = 1 is used when a switch requests an idle link leading to the corresponding downstream switch in setup phase. The Req = 1 is also kept during data transfer along the set up path. A Req = 0 denotes that the switch releases the occupied link. This code is also used in both the setup and the release phases. An Ans = 01 (Ack) means that the destination is ready to receive data from the source. When the Ans = 01

propagates back to the source, it denotes that the path is set up, then a data transfer can be started immediately. An Ans = 11 (nAck) is reserved for end-to-end flow control when the receiving circuit is not ready to receive data due to being busy with other tasks, or overflow at the receiving buffer, etc. An Ans = 11 (Back) means that the link is blocked. Dynamic Path

Setup to Support Path Arrangement

A dynamic path-setup scheme is the key point of the proposed design to support a runtime path arrangement when the permutation is changed. A path arrangement with full permutation consists of sixteen path setups, whereas a path arrangement with partial permutation may consist of a subset of sixteen path setups. The three-stage Close network C(m,n,p) is rearrangeable if m>=n. In the proposed network of C(4,4,4),m=n=4 so it is rearrangeable. There always exists an available path from an idle input leading to an idle output.The path setup completely searches all the possible paths within the set of path diversity between an idle input and idle output. Directly applying the search into rearrangeable C (4,4,4) shows that the path setup can always find an available path within the set of four possible paths between the input and the idle output. Based on this path-setup scheme, it is obvious that the path arrangement for full (as well as partial) permutation can always be realized in the proposed network with C(4,4,4) topology.

The following example describes how the path setup works to find an available path by using the set of path diversity shown in Fig. 2. It is assumed that a data from a source (e.g., an input of switch 01) is trying to set up a path to a target destination (e.g., an available output of switch 22). First, the data will non-repetitively try paths through the second-stage switches in the order of 10->11->12->13. Assuming that the link 01->10 is available, the probe first tries this link (Req=1) and then arrives at switch 10.

• If link 10-22 is available, the data arrives at switch 22 and meets the target output. An Ans=Ack then propagates back to the input to trigger the transfer phase.

• If link 10-22 is blocked, the data will move back to switch 01 (Ans=Back) and link 01-10 is released (Req=0). From switch 01, the data can then try the rest of idle links leading to the second-stage switches in the same manner. By means of moving back when facing blocked links and trying others, the data can dynamically set up the path in runtime in a conflict-avoidance manner.

Switching Node Designs

Three kinds of switches are designed for the proposed on-chip network.These switches are all based on a common switch architecture shown in Fig. 3. This common architecture has basic components: INPUT CONTROLs (ICs), OUTPUT CONTROLs (OCs) and an ARBITER.

The ARBITER has two functions: first, cross-connecting the Ans_Outs and the ICs through the Grant bus, and second, as a referee for the requests from the ICs. When an incoming data arrives at an input, the corresponding IC observes the output status through the Status bus, and requests the ARBITER to grant it access to the corresponding OC through the Request bus. When accepting this request, the ARBITER cross-connects the corresponding Ans_Out with the IC through the Grant bus with its first function. With the second function, the ARBITER, based on a pre-defined priority rule, Input Data resolves contention when several ICs request the same free output. After this resolution, only one IC is accepted, whereas the rest are answered as facing a blocked link (i.e., similar to receiving an Ans = Back).

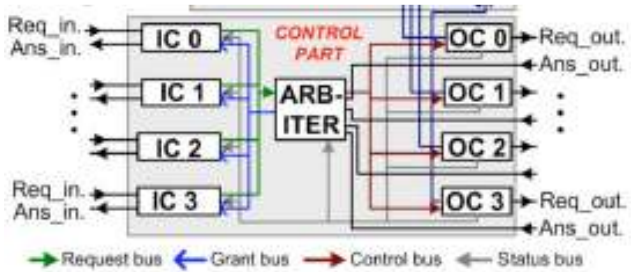

Figure 3: Common switch architecture

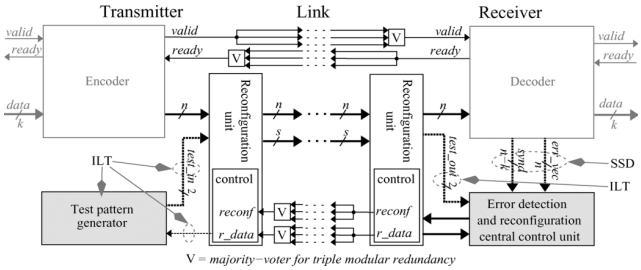

Figure 4: Reconfigurable link system. The parts required only for ILT or SSD implementation are marked in the figure.

The IC is implemented with finite-state machine (FSM). The operation of the switches are controlled according to this FSM implementation in the ICs. In order to support the path setup, ICs are implemented according to its switch stage.The data send is of 4 bits. In the first stage, the switch tries the free outputs in a non-repetitive manner (e.g., outputs 0->1->2->3). This implementation avoids repetitively searching the same path that may result in a live-lock. Depending on the availability of the desired output or the feedback (i.e., the signal Ans) from the downstream switch, the IC in a given switch will change its FSM state and reply to the upstream switches accordingly. The control part of switches performs the dynamic path setup. This meets the target of designing the circuit-switched switches to support path setup in C(4,4,4) network.

Permanent-Error Correction

Permanent-error correction in on-chip links using spare wires is a two-step process. First, the permanent error must be detected; then, the link must be reconfigured to avoid transmit ting over the faulty wire. The proposed adaptive link framework is shown in Fig. 1 and consists of a transmitter, a link, and a receiver. The incoming -bit-wide data word is encoded in the transmitter to a codeword of width , which is transmitted through the link and decoded in the receiver . The decoder is responsible for correcting any errors and outputs the original k–bit data word. A number of spare wires are available. Reconfiguration units at the transmitter and receiver determine which of the n + s lines carry data and which are left idle. The reconfiguration control units pass

reconfiguration information between the receiver and transmitter and synchronize reconfiguration. The error detection and reconfiguration central control unit detects permanent errors and initiates reconfiguration. The inputs to this unit depend on which detection method is used. For the SSD method, the syndrome (Synd) and error vector (err_vec) from the decoder are needed. For the ILT method, test outputs (test_out) from the spare wires under test are needed. The ILT method requires a test pattern generator (TPG) block and test inputs (test_in) to produce test signals.

We apply our techniques to permanent and intermittent errors in the link; the logic units are assumed to function correctly. Two methods are presented here, namely, the ILT method and the SSD method[29].

ILT Method

The proposed ILT method sequentially routes data from normal interconnects to a set of available spare wires, allowing tests for intermittent and permanent faults. This is achieved during normal operation, without interrupting data transmission, by making use of the reconfiguration system. To protect against runtime permanent errors, the ILT is run periodically, with a period that can be shortened to improve error resilience or increased for energy efficiency.

1) ILT Procedure: To begin the test TPG issues a series of test patterns using the test_in signal. The ILT control unit compares the received test_out signal to a lookup table(shown in Table II) to determine if there is a permanent error in that pair of wires. The lookup table indicates which line(s) need(s) to be flagged as erroneous. Note that, during each test, wires that were flagged as faulty are retested to prevent intermittent errors from wasting wire resources[30].

Faults are injected at the interconnects. Stuck-at fault mode is mainly chosen for identification of faults in the proposed network. Data is compared and from details in table error type is chosen.

SSD Method

Syndrome decoding is a common technique for decoding linear block codes. The syndrome is calculated by matrix multiplication s=u*(Transpose of H) , where u is a received code word vector of length n(u = c+e, where c is a transmitted code word and e is an error vector, both of length n),Transpose of H is the transpose of (n – k)*n the parity-check matrix, and s is a syndrome vector of length n-k. The syndrome gives the minimum weight error vector index, so the error vector e can easily be determined. The correction is done by c = u+e, eliminating the error from the received data word.

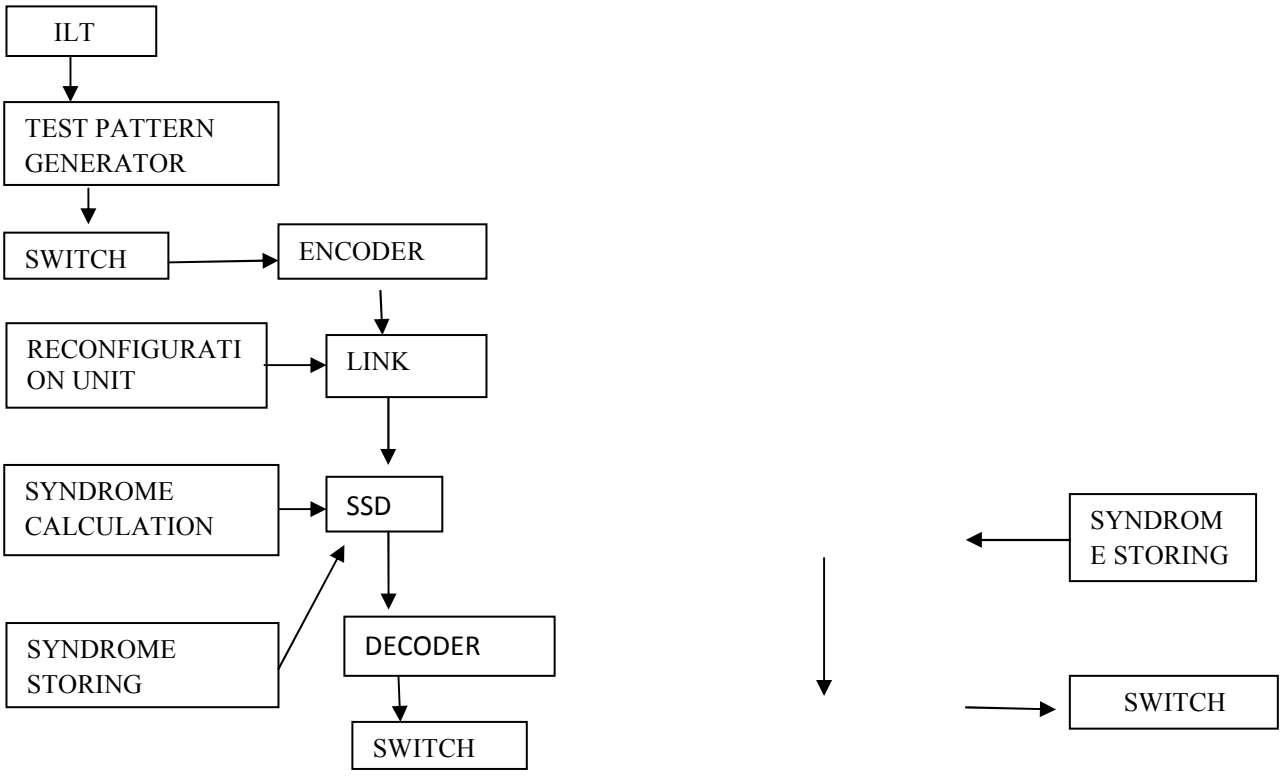

Figure 5: Self-adaptive on-chip network

The basic idea behind SSD is that the error syndrome of an error control code contains information about the errors of a received code word. If the syndromes of a number of consecutive received code words are the same, then it can be concluded that there is a permanent error in the link. The error location can be extracted from the syndrome using the normal decoding procedure. The effectiveness of this approach comes from the fact that it takes advantage of the code and decoder already present at the system. If there are more errors in the link than the error correction code is capable of correcting, the syndrome will be decoded incorrectly, providing a wrong error location.

1) Encoder and Decoder: The encoder and decoder are responsible for implementing the tolerance against transient

faults. The encoder calculates check bits that are transmitted together with the data word over the link, and these check bits are used in the decoder to detect and correct possible errors. The reconfiguration system is designed as a separate layer from the underlying data transmission. This means that there are no ECC requirements for the reconfiguration system to be able to bypass permanent errors.

2) Reconfiguration Units: The function of the reconfiguration unit is to route the data. The number of spare wires has an effect not only on the permanent- error tolerance

Table 1: Characteristics of the Designed Circuits

	Self-Adaptive	Encoder	Decoder	Permutation N/W
Min Period	9.654 ns	-	-	9.895ns
Max Frequency	103.584 MHz	-	-	101.059MHz
JTAG Gates for IOBs	34	1,632	1,632	1,632
Equivalent Gate count for Design	4,987	4,987	4,987	4,987

but also on the complexity of the reconfiguration units. The spare wires are used to replace faulty wires. The error detection circuits as part of the error detection and reconfiguration central control unit provide the location of the erroneous wire, which should then be bypassed. The control parts of the reconfiguration units at each end of the link are used to transmit the reconfiguration information from the receiver to the transmitter[31].

Self-adaptive On-chip Permutation Network Design

Test pattern generator issues a series of test patterns. This is passed through the switch in the on-chip permuatation network design. The encoder calculates check bits that are transmitted together with the data word over the link. Thus 7 bit data is given from the encoder (shown in Fig.5.) through the interconnecting link. This is then given to the SSD system where syndrome calculation and syndrome storing take place. The syndrome gives the minimum weight error vector index. The error location can be extracted from the syndrome using the normal decoding procedure. Thus extracts 4 bit data and given to the switch in following stage.

RESULTS

The analysis was performed by synthesizing the design. Area and speed results are presented in Table I. Maximum combinational path delay is calculated in each design. Minimum input arrival time before clock and maximum output required time after clock are also obtained in each synthesis reports. Decoding has two modes, i.e., one when there are no errors in the link, and the other when errors need to be corrected.

Table II: Calculation of Error Type

Test Vectors	Data Through Interconnect		Syndrome		Error Type
	N	S	Sn	Ss	
1000	1000110	1000110	000	000	No Error
0100	0100101	0100101	000	000	No Error
0010	0010011	0010011	000	000	No Error
0001	0001111	0001111	000	000	No Error
1000	1001110	1000110	111	000	Permanent
0100	0100101	0100101	111	000	Permanent
0010	0011110	0010110	111	000	Permanent
0001	0000111	0001111	111	000	Permanent
1000	1001110	1000110	000	000	Intermediate
0100	0101101	0100101	111	000	Intermediate
0010	0011011	0010011	111	000	Intermediate
0001	0001111	0001111	111	000	Intermediate Error

Area values include each system component and the registers required. Energy values were calculated from the average power and simulation time. The realizations use clock gating to reduce power and energy consumption. This can be seen from the standby power consumptions, which are measured when no data are flowing through the system but the clock is not stopped. Standby power consumptions are only a few milliwatts, a small fraction of the values during operation.The correctness of the reconfiguration procedure was validated with a series of simulations with both reconfigurable systems.Fig. 11 shows a waveform for a four input four output switch in an on-chip permutation network. Fig. 12 shows Simulation waveforms indicating intermediate error in self-adaptive network.

DISCUSSION

The achieved compactness of the proposed networks suggests that stacking multiple networks is feasible. Besides increasing bandwidth, the stacking can enable other benefits for further considerations. For example, to support simultaneous (partial) permutations path setups can be launched in parallel for speeding up. It is noted that the data delivered in the proposed network is guaranteed due to the use of circuit switching

whereas this feature is not clearly visible with the packet-switching approaches. Another example, assuming that a MPSoC is computing under a (standard) full permutation is that it then needs to switch to another permutation. A fast or even zero switching time can be achieved with stacking if a standby network is being rearranged in parallel with the current network's operation and is ready for the runtime switching. Regarding system scalability, the Clos topology is scalable as used in macro commercial systems. The proposed path-setup scheme performs in distribution, thereby suggesting scalability in terms of computing the guaranteed routes in runtime, compared to static (pre-computed) or centralized approaches. However, a runtime path-arrangement optimization and physical design issue for the scaled networks need more considerations in future researches.

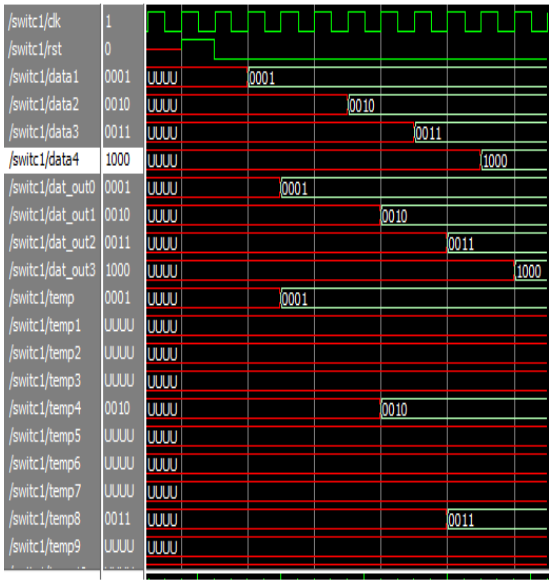

Figure 6: Waveform for a four input four output switch in on-chip permutation network

SSD always requires an ECC to be used on the link; however, ILT can be used on links without any ECC and, thus, without additional clock cycles. This could be useful, for example, in NoCs where end-to-end transient error protection is used or in media applications where small periods of data corruption are acceptable. In many cases, there are buffers at the inputs and outputs of the link (e.g., buffering at the routers of an NoC). This buffering capacity can be replaced by the buffer stages introduced to the transmitter and receiver when an ECC is used. Using these preexisting buffers, we would not need any additional clock cycles, and the latency increase would be eliminated.

Optimal selection of the number of spare wires is nontrivial. With SSD, high does not help if the correction capability of the underlying code is too low. ILT could be used with high, but increasing will also increase the delay of the reconfiguration units. The combination of the two presented detection methods could achieve additional benefits beyond those discussed in this paper. If the SSD method were the primary detection method, ILT properties could be used to further test wires declared erroneous. This would make it possible to return wires to normal usage if an error was intermittent. The SSD method could also be used to trigger an ILT test round, thus minimizing the overhead of unnecessary test rounds. The combination of these two schemes will be examined.

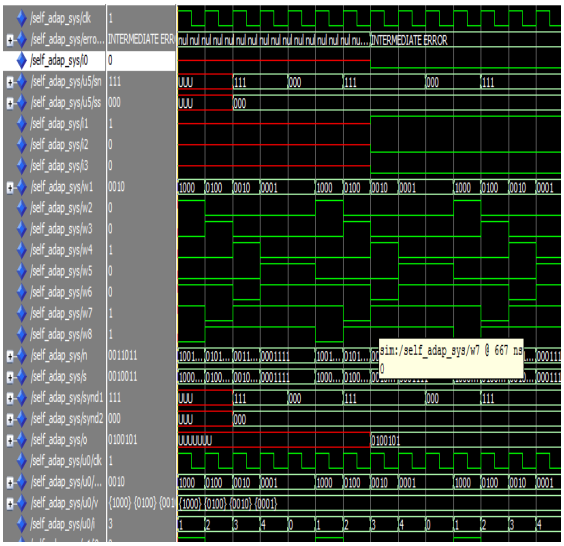

Figure 7: Simulation waveforms indicating intermediate error in self-adaptive network

CONCLUSION

This paper has presented an on-chip network design supporting traffic permutations in MPSoC applications. By using a circuit-switching approach combined with dynamic path-setup scheme under a Clos network topology, the proposed design offers arbitrary traffic permutation in runtime with compact implementation overhead. A complete reconfigurable system utilizing spare wires to replace erroneous wires and enabling reconfiguration without interfering with data transmission has been presented. Two methods for error detection have been evaluated, namely, rotating ILT and SSD.

This paper shows that protection against permanent errors can be achieved using spare wires with smaller penalties than using complex coding schemes.

ACKNOWLEDGEMENT

The authors would like to thank Dept. of ECE for supporting all along to the successful completion of the project.

REFERENCES

1. Pham PH, Song J, ParkJ, And KimC,Design and implementation of an on-chip permutation network for multiprocessor system on chip in IEEE transactions on very large scale integration (VLSI) systems, 2013; 21: 1.
2. Lehtonen T, Wolpert D, Liljeberg P, Plosila J, And Ampadu P, Self adaptive system for addressing permanent errors in on-chip interconnects",in ieee

transactions on very large scale integration (VLSI) systems, 2010; 18: 4.
3. Borkar S, Thousand core chips-A technology perspective, inACM/IEEE Design Autom. Conf. (DAC), 2007; 746–749.
4. Pham PH, Mau P, and Kim C, A 64-PE folded-torus intra chipcommunication fabric for guaranteed throughput in network on chip based applications, in Proc. IEEE Custom Integr. Circuits Conf.(CICC), 2009; 645–648.
5. Moussa H, Baghdadi A, and Jezequel M, Binary de Bruijn on network for a flexible multiprocessor LDPC decoder, in Proc. ACM/IEEE Design Autom. Conf. (DAC), 2008; 429–434.
6. H. Moussa, O. Muller, A. Baghdadi, and M. Jezequel, "Butterfly benes-based on-chip communication networks for multiprocessor turbo decoding," in Proc. Design, Autom. Test in Euro. (DATE).
7. Ludovici D, Gilabert F, Medardoni S, Gomez C, Gomez ME, Lopez P, Gaydadjiev GN, and Bertozzi D, Assessing fat-tree topologies for regular network-on-chip design under nanoscale technology constraints, in Proc. Design, Autom. Test Euro. Conf. Exhib. (DATE), 2009; 562–565.
8. Yang Y, and Wang J, A fault-tolerant rearrangeable permutation network, IEEE Trans. Comput, 2004; 53(4):414–426.
9. Gaughan PT, and Yalamanchili S, A family of fault-tolerant routing protocols for direct multiprocessor networks, IEEE Trans. ParallelDistrib. Syst., 1995; 6(5):482–497, 1995.
10. Bene˘s VE, Mathematical theory of connecting networks andtelephone traffic. New York: Academic Press, 1965.
11. Bertozzi D, Benini L, and De Micheli D, Error control schemes for on-chip communication links: The energy-reliability tradeoff,IEEETrans. Comput-Aided Des. Integr. Circuits Syst., 2005; 24(6):818–831.
12. Li L, Vijaykrishnan N, Kandemir M, and Irwin MJ, "Adaptive error protection for energy efficiency," in Proc. ICCAD, San Jose, CA, 2003; 2–7.
13. Murali S, Theocharides T, Vijaykrishnan N, Irwin MJ, Benini L,and Micheli GD, Analysis of error recovery schemes for networks on chips,IEEE Des. Test Comput., 2005; 22(5):434–442.
14. Rossi D, Angelini P, and Metra C, Configurable error controlscheme for NoC signal integrity, in Proc. 13th IEEE IOLTS, Crete, Greece, 2007; 43–48.
15. Sridhara SR and Shanbhag NR, Coding for system-on-chip networks: A unified framework,IEEE Trans. Very Large Scale Integr(VLSI) Syst., 2005;13(6),:655–667.
16. Yu Q and Ampadu P, Adaptive error control for reliable systems-onchip, in Proc. IEEE ISCAS, Seattle, WA, 2008; 832–835.
17. Zimmer H and Jantch A, A fault model notation and error-controlscheme for switch-to-switch buses in a network-on-chip, in Proc. 1st IEEE/ACM/IFIP CODES+ISSS, Newport Beach, CA, 2003; 188–193.
18. Lehtonen T, Liljeberg P, and Plosila J,Online reconfigurable selftimed links for fault tolerant NoC," VLSI.,2007;1–13.
19. Ziemer R and Peterson R, Introduction to Digital communication,2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2001.
20. Ejlali A, Al-Hashimi BM, Rosinger P, and Miremadi SG, Joint consideration of fault-tolerance, energy-efficiency and performance in on-chip networks," in Proc. DATE, Nice, France, 2007;1–6.
21. Shang L, Peh LS, and Jha NK, Dynamic voltage scaling with links for power optimization of interconnection networks, in Proc. 9th Int.Symp. HPCA, Anaheim, CA. 2003;91–102.
22. Johnson B, Design and analysis of fault tolerant digital systems.Boston, MA: Addison-Wesley, 1989.
23. Hanchek F and Dutt S, Methodologies for tolerating cell and interconnect faults in FPGAs,IEEE Trans. Comput., 1998;47(1):15–33.
24. Yu AJ and Lemieux GF, Defect-tolerant FPGA switch block and connection block with fine-grain redundancy for yield enhancement, in Proc. Int. Conf. Field Program. Logic Appl., Tampere, Finland, 2005; 255–262.
25. Refan F, Alemzadeh H, Safari S, Prinetto P, andNavabi Z, Reliability in application specific mesh based NoC architectures, in Proc.IEEE IOLTS, Rhodes, Greece, 2008; 207–212.
26. Grecu C, Ivanov A, Saleh R, and Pande PP, Testing network- onchip communication fabrics, IEEE Trans. Comput.-Aided Des. Integr.Circuits Syst., 2007; 26(12): 2201–2214.
27. Reick K, Sanda PN, Swaney S, Kellington JW, Mack MJ, Floyd MS, and Henderson D, Fault-tolerant design of the IBM Power6 microprocessor,IEEE Micro, 2008; 28(2):30–38.
28. Shin J, Kim H, and Kang S, At-speed boundaryscan interconnect testing in a board with multiple system clocks, in Proc. DATE, Munich, Germany, 1999; 473–477.
29. Bloom DM, Probabilities of clumps in a binary sequence (and how to evaluate them without knowing a lot),Math. Mag., 1996; 69(5): 366–372.
30. Weisstein EW, CRC Concise Encyclopedia of Mathematics, 2nd. ed. Boca Raton, FL: Chapman & Hall/CRC, 2003.
31. Lu Z, Huang W,Stan MR, Skadron K, and J. Lach J, Interconnect lifetime prediction for reliability aware systems,IEEE Trans. VeryLarge Scale Integr. (VLSI) Syst., 2007; 15(2): 159–172.

Source of support: Nil, Conflict of interest: None Declared