A reconfigurable routing algorithm for a fault-tolerant 2D-Mesh Network-on-Chip

Zhen Zhang, Alain Greiner, Sami Taktak

A Reconfigurable Routing Algorithm for a Fault-Tolerant 2D-Mesh Network-on-Chip

Zhen Zhang
zhen.zhang@lip6.fr

Alain Greiner
alain.greiner@lip6.fr

Sami Taktak
sami.taktak@lip6.fr

Univ Pierre et Marie Curie & LIP6-SOC
4, Place Jussieu, 75252 Paris, France

ABSTRACT

In this paper we present a reconfigurable routing algorithm for a 2D-Mesh Network-on-Chip (NoC) dedicated to fault-tolerant, Massively Parallel Multi-Processors Systems on Chip (MP2-SoC). The routing algorithm can be dynamically reconfigured, to adapt to the modification of the micro-network topology caused by a faulty router. This algorithm has been implemented in a reconfigurable version of the DSPIN micro-network, and evaluated from the point of view of performance (penalty on the network saturation threshold), and cost (extra silicon area occupied by the reconfigurable version of the router).

Categories and Subject Descriptors

B.8.1 [**PERFORMANCE AND RELIABILITY**]: Reliability, Testing, and Fault-Tolerance; C.1.2 [**PROCESSOR ARCHITECTURES**]: Multiprocessors—*Interconnection architectures*

General Terms

Design, Algorithms, Reliability

Keywords

2D-Mesh NoC, fault-tolerant, routing algorithm, MP2-SoC, reconfiguration, DSPIN

1. INTRODUCTION

The Network-on-Chip (NoC) has been recognized to solve the bandwidth bottleneck, when interconnecting a huge number of IP-cores in Massively Parallel Multi-Processors Systems on Chip (MP2-SoC). According to the prediction from an INTEL commentator in [10]: *"within a decade we will see 100 billion transistor chips"*, namely, the NoC-based MP2-SoC will integrate thousands of IP-cores. However, *"20 billion of those transistors will fail in manufacture and a further 10 billion will fail in the first year of operation"*. Thus, a 20%-30% device failure rate means that the fault-tolerant approach must be considered in MP2-SoC design.

As any MP2-SoC architecture will contain a large number of replicated components, a simple fault-tolerant approach is to deactivate the defective components (such as a processor core or an embedded RAM), once it has been detected as a failure, and to map the software application on the remaining hardware components. Unfortunately, this simple deactivation approach can't deal with a faulty router in the NoC itself. In order to save silicon area, and to minimize the network latency, most NoCs use dedicated routing algorithms, taking advantage on the regular micro-network topology. Once a router has failed, the deactivation of the faulty router is not enough, as the micro-network topology is modified and irregular. If the routing algorithm continues to route the packets toward the faulty router, the micro-network will block. Therefore, the routing algorithm must be modified (reconfigured) to adapt to the modification of the micro-network topology. To realize such reconfiguration, it is necessary to solve 3 problems:

A	The faulty router must be detected by an appropriate built-in self-test mechanism.
B	A fault-tolerant, distributed, reconfigurable routing algorithm, must be defined, and implemented in the routers.
C	A robust configuration bus must be implemented in the hardware to distribute the configuration information to the routers.

In this paper, we address problem B, and we present a fault-tolerant, distributed, reconfigurable routing algorithm that can be used in any 2D-Mesh NoC. This algorithm has been implemented in a reconfigurable version of the DSPIN [7, 15] micro-network. Problems A and C are not addressed in this paper.

Related results: During the last two decades, a lot of approaches have been published about fault-tolerant wormhole routing algorithms (a review of wormhole routing algorithms can be found in [14]) for 2D-Mesh network. These approaches could be split in two classes: the virtual channel (VC) model and the turn model.

The VC-based fault-tolerant routing algorithms in [2–4, 13, 16] allow a single physical channel to be shared by multiple transactions, using some form of time multiplexing [5]. According to Duato's results in [8, 9], a VC-based fault-tolerant routing algorithm can route the packets around a faulty router or a faulty region (including multiple faulty routers) using different VCs to avoid deadlocks. In a NoC, the hardware cost of the router must be kept very low. We

believe that the high complexity and large area cost associated to the VC-based approach are not compatible with this low-cost constraint.

The turn model is originated from Glass and Ni's studies in [11]. They published three adaptive routing algorithms: West-First (WF), North-Last (NL) and Negative-First (NF). These routing algorithms eliminate deadlocks without adding VC, by prohibiting some global turns as shown in Figure 1 (<A>, <B>, <C>). However, they can't deal with some one-faulty-router topologies as shown in Figure 1 (<D>).

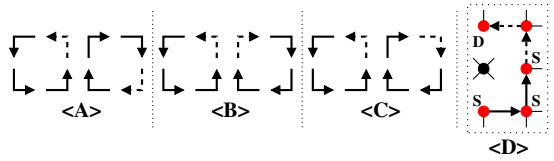

Figure 1: Six global turns allowed (solid lines) and two global turns prohibited (dotted line) by WF (<A>), by NL (<B>) and by NF (<C>). The prohibited turn in these three routing algorithms prohibites also the packet from S to D (X is a faulty router) as shown in <D>.

Glass and Ni proposed a turn-based fault-tolerant routing algorithm in [12] that is based on modification of the Negative-First routing algorithm. It can deal with any one-faulty-router topology. But each routing function depends on the coordinates (Y,X) of the router, the packet destination, the input channels, and the size of the $N \times M$ mesh. This last dependency is the weakest point, as the hardware complexity associated to this modified routing algorithm depends on the mesh size. Moreover, this routing algorithm copes with one faulty router, but cannot handle one faulty region containing several faulty routers.

Finally, a fault-tolerant wormhole routing algorithm for a 2D-Mesh NoC must respect the following constraints:

Low cost	The hardware cost resulting from the reconfigurability must be a small percentage of the total router silicon area.
Generic	The reconfigurable routing algorithm must handle any one-faulty-router (or one-faulty-region) topology.
Scalable	The reconfiguration hardware must be independent on the 2D-Mesh size.
Deadlock free	Any reconfigured routing algorithm must be deadlock free.
Deterministic	The resulting routing algorithm must guaranty the In-order delivery property

Organization of the paper: The section 2 presents a typical 2D-Mesh NoC: DSPIN. The section 3 explains the principles of our reconfigurable routing algorithm. The section 4 presents a reconfigurable version of the DSPIN router, and the section 5 contains experimental results related to the performance (penalty on the network saturation threshold), and to the hardware cost (extra silicon area occupied by the reconfigurable version of the router).

2. A TYPICAL 2D-MESH NOC ⇒ DSPIN

DSPIN & MP2-SoC: The DSPIN micro-network (Distributed Scalable Predictable Interconnect Network) was designed by the LIP6 laboratory and physically implemented by ST Microelectronics. It is a typical 2D-Mesh NoC, supporting MP2-SoCs architectures, and the GALS (Globally Asynchronous Locally Synchronous) approach. Each subsystem is a synchronous domain. In the following, we call "cluster" such a subsystem, containing one or several processor cores, a local interconnect, a network interface controller (NIC), and two routers. In order to avoid deadlocks in commands/responses traffic, each cluster contains two independent routers (as shown in Figure 2 (<A>)) implementing two separated subset-networks for commands and responses. Each subset-network has a 2D-Mesh topology as shown in Figure 2 (<B>).

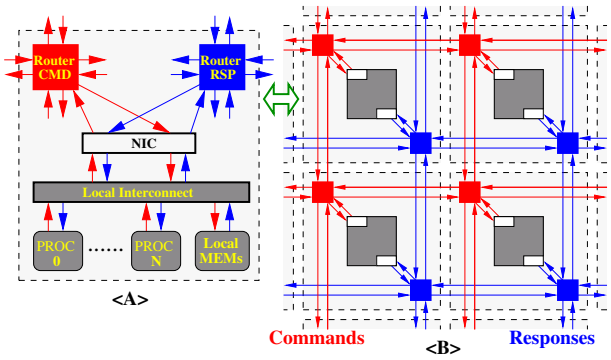

Figure 2: <A> shows a generic DSPIN-based MP2-SoC. In such a DSPIN-based MP2-SoC, a dotted rectangle means a generic DSPIN-based MP2-SoC's cluster as shown in <B>.

Routing algorithm: DSPIN is a packet-switching network: a packet is broken into smallest flow control unit called flits. The first flit is the head flit and the last is the tail flit. The head flit includes the destination "cluster address" defined in absolute coordinates Y and X. Once the head flit of a packet is received by a router, the destination field is analyzed and the flit is routed to the corresponding output port. The rest of the packet is also routed to the same port until the tail flit.

DSPIN uses a determinist and deadlock free routing algorithm: X-First [6]. With this routing algorithm, the packets are firstly routed on the X direction and then on the Y direction. The X-First routing function depends on the coordinates of the router (Y_Local, X_Local) and the coordinates of the destination ($Y\_Destination, X\_Destination$) as presented in Listing 1.

Listing 1: The X-First routing function source code in SystemC

```
if ( X_Destination > X_Local )                    1
  Out = EAST;                                     2
else if ( X_Destination < X_Local )               3
  Out = WEST;                                     4
else if ( Y_Destination > Y_Local )               5
  Out = NORTH;                                    6
else if ( Y_Destination < Y_Local )               7
  Out = SOUTH;                                    8
else                                              9
  Out = LOCAL;                                    10
```

One important feature of the X-First routing algorithm is the following: The packet path from a node (y, x) to the node (y', x') is a ***Unique Path***:

$$L = \{R_0(y,x), ..., R_{|x'-x|}(y,x'), ..., R_{|y'-y|+|x'-x|}(y',x')\}$$

The router's architecture: As shown in Figure 3, the DSPIN router is composed of 5 modules (North, East, South, West & Local), and the DSPIN router is not a full crossbar: some interconnections between modules have been removed, according to the X-First routing algorithm (NORTH → WEST, NORTH → EAST, SOUTH → WEST and SOUTH → EAST), reducing the complexity of the multiplexers in the EAST and WEST modules.

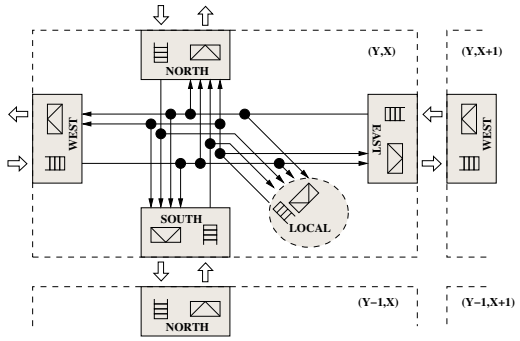

Figure 3: A generic DSPIN router's architecture.

The faulty router's model: We make the assumption that a faulty router can be detected by a dedicated build-in self-test mechanism, which won't be described in this paper. Even if the DSPIN router is architecturally composed of 5 modules, when any module or interconnection of a router is detected as faulty, the entire router will be considered as faulty. Moreover, if one of the two routers in a cluster is detected as faulty, it breaks the commands/responses protocol. So the other router of this cluster must be also considered as faulty, the corresponding cluster must be considered as a "hole" in the mesh. All components in the hole must be deactivated. As the Unique Path has been broken by a hole, the dynamic reconfiguration mechanism must restore the broken Unique Path.

3. THE RECONFIGURABLE ROUTING ALGORITHM

The main idea of the deterministic, fault-tolerant, distributed, reconfigurable routing algorithm is to route the packets through a cycle free contour surrounding a faulty router, so as to restore all broken Unique Paths.

Definition 1: Neighbors.
In a 2D-Mesh, a node (Y,X) has 4 direct neighboring nodes (N,S,W,E) and 4 indirect neighboring nodes (NE,NW,SE,SW). We call those 8 nodes the neighbors, as shown in Figure 4.

Definition 2: Natural contour.
The neighbors of a hole (Y,X) define a natural contour as shown in Figure 4. It separates the network into two parts: normal part A and defective part B.

Definition 3: Cycle free contour.
The location of a single hole has $N \times M$ possibilities in a $N \times M$ 2D-Mesh. Thus, a natural contour has 9 possible shapes corresponding to 9 types of locations: at each corner, at each side and at other positions, as shown in Figure 5.

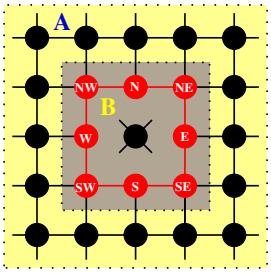

Figure 4: A generic hole's neighbors and the natural contour.

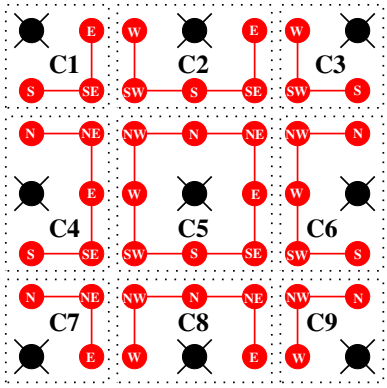

Figure 5: 9 natural contours.

Following Dally's condition in [6], the *channel dependency graphs* (CDG) can be used to prove the 8 natural contours (C1,...,C4,C6,..,C9) to be deadlock free. In a CDG, the nodes are the communication channels (not the routers). There is a directed edge from node (i) to node (j) when (i) is an input channel for router R, (j) is an output channel for router R, and the routing function associated to R defines a possible path from (i) to (j). The natural contour C5 is NOT deadlock free, as there is 2 cycles in C5's CDG, as shown in Figure 6.

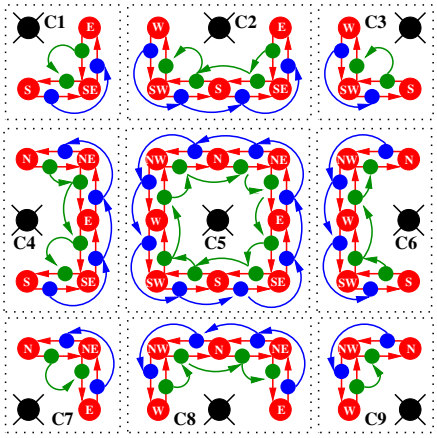

Figure 6: The CDGs of 9 natural contours. 2 cycles are found in the C5's CDG, so C5 can introduce deadlock.

We adopt a turn-based fault-tolerant approach to break the 2 cycles in C5's CDG, by prohibiting the two NE turns, as shown in Figure 7. As a result, we defined 9 cycle free con-

tours, corresponding to the 9 possible locations for a faulty router.

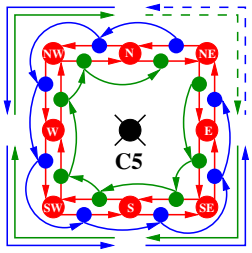

Figure 7: The two turns prohibited (dotted line) in C5's NE can break the 2 cycles.

Definition 4: Modified routing function.

We define a modified routing algorithm for all routers that are part of a cycle free contour around a faulty router, in order to restore all broken Unique Paths. In case of a faulty router, we make the assumption that all components in the cluster have been deactivated, and the faulty cluster is neither the source nor the destination of any packet. Therefore, the 8 paths Li (corresponding to the X-First routing function) broken by this hole can be explicitly listed in Table 1. As described in Figure 8, for each broken path Li, the modified routing function defines a new path $NewLi$ also listed in Table 1.

Table 1: the 8 Li are restored by the 8 $NewLi$

L_1	$\{R_W, R_x, R_N\}$
$NewL_1$	$\{R_W, R_{NW}, R_N\}$
L_2	$\{R_E, R_x, R_N\}$
$NewL_2$	$\{R_E, R_{SE}, R_S, R_{SW}, R_W, R_{NW}, R_N\}$
L_3	$\{R_W, R_x, R_S\}$
$NewL_3$	$\{R_W, R_{SW}, R_S\}$
L_4	$\{R_E, R_x, R_S\}$
$NewL_4$	$\{R_E, R_{SE}, R_S\}$
L_5	$\{R_W, R_x, R_E\}$
$NewL_5$	$\{R_W, R_{SW}, R_S, R_{SE}, R_E\}$
L_6	$\{R_E, R_x, R_W\}$
$NewL_6$	$\{R_E, R_{SE}, R_S, R_{SW}, R_W\}$
L_7	$\{R_N, R_x, R_S\}$
$NewL_7$	$\{R_N, R_{NW}, R_W, R_{SW}, R_S\}$
L_8	$\{R_S, R_x, R_N\}$
$NewL_8$	$\{R_S, R_{SW}, R_W, R_{NW}, R_N\}$

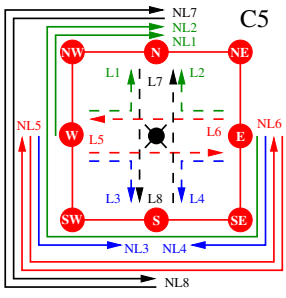

Figure 8: the 8 Li (dotted line) broken by a hole are restored by the 8 $NewLi$ (solid lines).

As shown in Figure 9, a similar approach can be defined for the 8 others cycle free contours.

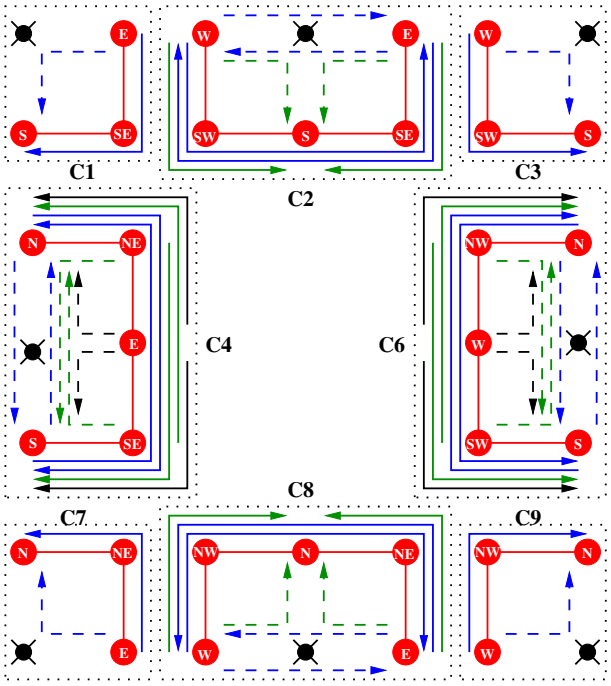

Figure 9: The broken Li and $NewLi$ in other cycle free contours.

4. HARDWARE IMPLEMENTATION

In a 2D-Mesh, a given router R can be in 9 different situations: If none of the 8 neighboring routers is faulty, R is configured as NORMAL, implementing the classical X-First routing function. If one of the neighbors is faulty, R is part of a cycle free contour, and must be configured accordingly (N_OF_x, S_OF_x, E_OF_x, W_OF_x, NE_OF_x, NW_OF_x, SE_OF_x, SW_OF_x), implementing a modified routing function. To implement the reconfigurable routing algorithm, two main modifications have been introduced in the DSPIN router micro-architecture:

- The interconnections NORTH → WEST, NORTH → EAST, SOUTH → WEST and SOUTH → EAST must be restored, and the multiplexers in the EAST and WEST modules must have 4 inputs, as described in Figure 10.

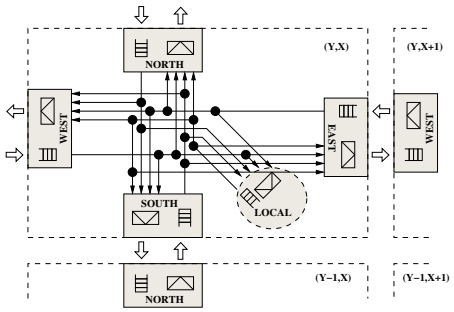

Figure 10: A generic architecture of reconfigurable DSPIN's router.

- As there is 9 possible configurations for a given router, the configuration information must be stored in a 4 bits register, and the X-First routing function must be modified to introduce a dependency on the value stored in the configuration register, as shown in Listing 2.

Listing 2: The X-First routing function modified by the configuration register (SystemC code)

```
if( X_Destination > X_Local ){
    if( REGISTER == NE_OF_x ||
        REGISTER == E_OF_x  ||
        REGISTER == SE_OF_x ||
        REGISTER == S_OF_x  ||
        REGISTER == NORMAL )
        OUT = EAST;
    else if( REGISTER == N_OF_x ){
        if( Y_Local == 1                ||
            X_Local == 0                ||
            Y_Destination >= Y_Local    ||
            X_Destination > X_Local + 1 )
            OUT = EAST;
        else
            OUT = WEST;
   }else if( REGISTER == NW_OF_x ){
        if( Y_Local == 1                ||
            Y_Destination >= Y_Local    ||
            X_Destination > X_Local + 2 )
            OUT = EAST;
        else
            OUT = SOUTH;
   }else if( REGISTER == W_OF_x ){
        if( Y_Local == 0                ||
            Y_Destination > Y_Local )
            OUT = NORTH;
        else
            OUT = SOUTH;
   }else{
        if( Y_Destination <= Y_Local    ||
            X_Destination > X_Local + 1 )
            OUT = EAST;
        else
            OUT = NORTH;
   }
}else if( X_Destination < X_Local ){
    if( REGISTER == N_OF_x  ||
        REGISTER == NW_OF_x ||
        REGISTER == W_OF_x  ||
        REGISTER == SW_OF_x ||
        REGISTER == S_OF_x  ||
        REGISTER == NORMAL )
        OUT = WEST;
    else if( REGISTER == NE_OF_x ){
        if( X_Destination < X_Local - 1 ||
            Y_Destination >= Y_Local )
             OUT = WEST;
        else
             OUT = SOUTH;
   }else if( REGISTER == SE_OF_x ){
        if( X_Local == 1                &&
            Y_Destination > Y_Local + 1 )
            OUT = NORTH;
        else
            OUT = WEST;
   }else{
        if( Y_Local == 0                ||
            ( X_Local == 1              &&
              Y_Destination > Y_Local ) )
             OUT = NORTH;
        else
             OUT = SOUTH;
   }
}else if( Y_Destination > Y_Local ){
    if( REGISTER != S_OF_x )
        OUT = NORTH;
    else if( X_Local != 0 )
        OUT = WEST;
    else
        OUT = EAST;
}else if( Y_Destination < Y_Local ){
    if( REGISTER != N_OF_x )
        OUT = SOUTH;
    else if( X_Local != 0 )
        OUT = WEST;
    else
        OUT = EAST;
}else
    OUT = LOCAL;
```

This routing function has been analyzed from the point of view of deadlocks: For the defective part B, all contours are deadlock free. For the normal part A, the reference X-First routing algorithm is also deadlock free. In order to prove that this reconfigurable routing algorithm is deadlock free, we used the formal proof tool ODI [17], developed at LIP6. This tool is dedicated to deadlock analysis in packet switching networks. It is based on the analysis of "Strongly Connected Components" (SCC) of the Extended Dependency Graph defined by the micro-network topology on one hand, and by the routing algorithm on the other hand. Each router can have a different routing function, and the routing function depends on the destination defined in the packet header. This tool tries to build a sufficient condition proving the routing algorithm to be deadlock free. We have proved the proposed routing algorithm to be deadlock free in any one-faulty-router topology, for a 10×10 2D-Mesh.

5. EXPERIMENTAL RESULTS

Performance (penalty on the network saturation threshold): The cycle-accurate, bit-accurate SystemC simulation model of the DSPIN router has been modified to implement the reconfigurable routing algorithm described in section IV. We simulated a 2D-Mesh containing 5×5 clusters. Each cluster contains one traffic generator and one target. For each initiator, the offered load (defined as the ratio between the number of injected flits and the total number of cycles) can be precisely adjusted. The traffic has a uniform random distribution (each initiator sends packets to all targets). The packet length is 8. The average network latency is measured as the average number of cycles for a round trip from an initiator to a target, and back to the same initiator. If we plot the average latency versus the offered load, the saturation threshold is the maximal accepted load where the latency increases to infinity. We simulated all one-faulty-router topologies, and the Figure 11 presents the results for 5 cases: no hole, hole in (0,0), hole in (0,2), hole in (1,1), hole in (2,2).

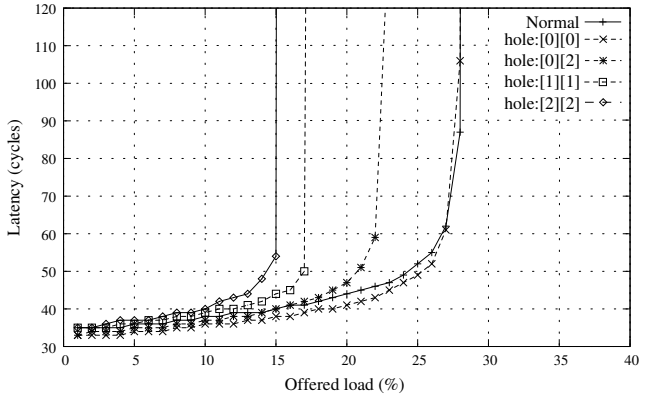

Figure 11: Some saturation thresholds in 5×5 2D-Mesh.

When the load is not too high, the impact of the modified routing algorithm on the average latency is negligible, but the saturation threshold can be strongly modified, when the hole is located at the center of the mesh.

Cost (extra silicon area): The synthesizable VHDL model of the DSPIN router has been modified to introduce the reconfigurable routing algorithm described in section IV. We used the SXLIB standard cell library [1] for a 90nm CMOS technology, and the Synopsys synthesis environment to evaluate the cost of the reconfigurability from the point of view of the silicon area.

DSPIN				
	Router Without FIFOs		Total With FIFOs	
	λ	mm^2	λ	mm^2
Original	1836600	0.015	7831600	0.063
Reconfigurable	2459250	0.020	8454250	0.068
Increase (area)	622650	0.005	622650	0.005
Increase (%)	33.90%		7.95%	

The router footprint is increased by only 8%. This is a very low cost, as the routers represent about 3% of the silicon area for a typical cluster. This numbers do not take into account the BIST logic for network testability.

6. CONCLUSION

We propose an ultra-low-cost reconfigurable routing algorithm supporting any one-faulty-router topology. It requires only a 4bits configuration register per router. It has been physically implemented in the DSPIN micro-network. The silicon area penalty is only 8% of the router footprint, and about 0.2% of the total chip area. The impact on the latency and saturation threshold has been evaluated. The reconfigurable routing algorithm is fully scalable. It has been demonstrated in the DSPIN micro-network, but can be used in any 2D-Mesh Network-on-Chip.

Moreover, this algorithm can be extended to one-faulty-region topology. The faulty region is a rectangle covering all faulty routers as shown in Figure 12. All internal clusters in this faulty region must be considered as faulty and deactivated.

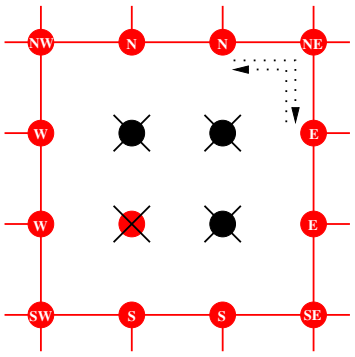

Figure 12: In a faulty region, a rectangular contour is built around this region.

In massively parallel multi-processors architecture, this reconfiguration capability can become mandatory to improve the yield issues.

7. REFERENCES

[1] Alliance CAD.

[2] R. Boppana and S. Chalasani. Fault-tolerant wormhole routing algorithms for mesh networks. *Computers, IEEE Transactions on*, 44(7):848–864, 1995.

[3] A. Chien and J. Kim. Planar-Adaptive Routing:Low-Cost Adaptive Networks for Multiprocessors. *JACM*, 42:91–123, 1995.

[4] C. Cunningham and D. Avresky. Fault-tolerant adaptive routing for two-dimensional meshes. *The 1 st IEEE Symposium on High-Performance Computer Architecture*, pages 122–131, 1995.

[5] W. Dally. Virtual-Channel Flow Control. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):194–205, 1992.

[6] W. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Transactions on Computers*, 36(5):547–553, 1987.

[7] DSPIN. *http://www.lip6.fr/Direction/2005-05-13-DSPIN.pdf*.

[8] J. Duato. A Necessary and Sufficient Condition for Deadlock-Free Adaptive Routing in Wormhole Networks. *IEEE Transactions on Parallel and Distributed Systems*, 6(10):1055–1067, 1995.

[9] J. Duato. A Theory of Fault-Tolerant Routing in Wormhole Networks. *IEEE Transactions on Parallel and Distributed Systems*, 8(8):790–802, 1997.

[10] S. Furber. Living with Failure: Lessons from Nature? *Proceedings of the Eleventh IEEE European Test Symposium (ETS'06)-Volume 00*, pages 4–8, 2006.

[11] C. Glass and L. Ni. The turn model for adaptive routing. *Proceedings of the 19th annual international symposium on Computer architecture*, pages 278–287, 1992.

[12] C. Glass and L. Ni. Fault-tolerant wormhole routing in meshes. *Fault-Tolerant Computing, 1993. FTCS-23. Digest of Papers., The Twenty-Third International Symposium on*, pages 240–249, 1993.

[13] D. Linder and J. Harden. An Adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes. *IEEE Transactions on Computers*, 40(1):2–12, 1991.

[14] L. Ni and P. McKinley. A Survey of Wormhole Routing Techniques in Direct Networks. *Computer*, 26(2):62–76, 1993.

[15] I. Panades, A. Greiner, A. Sheibanyrad, and G. STMicroelcctronics. A Low Cost Network-on-Chip with Guaranteed Service Well Suited to the GALS Approach. *Nano-Networks and Workshops, 2006. NanoNet'06. 1st International Conference on*, pages 1–5, 2006.

[16] C. Su and K. Shin. Adaptive fault-tolerant deadlock-free routing in meshes and hypercubes. *IEEE Transactions on Computers*, 45(6):666–683, 1996.

[17] S. Taktak, E. Encrenaz, and J. Desbarbieux. A Tool for Automatic Detection of Deadlock in Wormhole Networks on Chip. *High-Level Design Validation and Test Workshop, 2006. Eleventh Annual IEEE International*, pages 203–210, 2006.