Routing-path Tracking and Updating Mechanism in Reconfigurable Network-on-Chips

Thi-Thuy Nguyen, Thanh-Vu Le-Van, Hung K. Nguyen, Xuan-Tu Tran*
Key Laboratory for Smart Integrated Systems (SISLAB),
VNU University of Engineering and Technology (VNU-UET), Hanoi, Vietnam
{thuynguyen, levtvu, kiemhung, tutx}@vnu.edu.vn; (*) corresponding author

***Abstract*— As the rapid advancement in semiconductor technology leads to shrinking of transistor sizes and the integration scale is over one billion transistors, Reconfigurable Network-on-Chips becomes a new methodology providing adaptive infrastructure resources as well as flexible network protocols to adapt to dynamic environment. In this paper, we propose a routing-path tracking and updating mechanism for reconfigurable Network-on-Chips. The hardware architectures used to implement the proposed mechanism such as modified RNoC routers and Network Interfaces are introduced. With this routing-path tracking and updating mechanism, packet transmission delay can be reduced from 4.92% to 33.33% depending on the data structures.**

Keywords—Reconfigurable Network-on chip; packet delay

I. INTRODUCTION

The Network-on-chip (NoC) has become an emerging paradigm for on-chip communication of large complex system-on-a-chips with many prestigious advantages [1]. However, as the technology is shrinking and the need of applications increases rapidly, the current NoC models are not flexible enough to adapt to dynamic environments in which the main characteristics of system can be changed in run time [2] or there is an unexpected defect [3]. Therefore, reconfigurable NoCs, a new NoC platform which provides adaptive infrastructure resources and flexible network protocols, has being gotten the attention of researchers.

There existed many works on reconfigurable NoC solutions such as [4] providing reconfigurable topology, [5] introducing reconfigurable links. The Reconfigurable NoC (RNoC) platform developed in [6] provides a dynamic reconfigurable communication mechanism by adding a virtual port called RM port to the conventional router. For instance, when a packet is being transmitted to a prohibited router, the header flit or all flits of that packet are forwarded to the RM port instead of the targeted output port. At the RM port, the header flit is analyzed to create a new routing-path for the packet. However, the process at RM port takes time, causing an increase in packet transmission delay. This means that the time a packet occupies the network communication resources is lengthen. This may lead to the bad effect to the traffic of the NoC. To tackle this potential problem, we propose a routing-path tracking and updating mechanism aims at reducing packet transmission delay.

The rest of this paper is organized as follows. In Section II, the proposed routing-path tracking and updating mechanism is introduced. Then, the design of modified RNoC router and Network Interface (NI) are described in Section III. Section IV presents the evaluation method and the implementation results. Finally, conclusions and remarks are given in Section V.

II. ROUTING-PATH TRACKING AND UPDATING MECHANISM

As mentioned above, to address with the increase in packet transmission delay caused by the process in RM port, a routing-path tracking and updating mechanism is proposed. When a packet is transmitted over the network, its routing-path will be tracked, stored and provided at the destination router. Whenever the routing-path is changed to avoid prohibited routers, the new routing-path will be processed at the destination router. Then the information of new routing-path is packaged and sent back to the source router in an updating packet. When updating packet is received at the source router, the old routing path in the routing table will be replaced by the new one. As a result, after updating the routing table, packets will be transmitted with new routing-path which allows passing prohibited routers without going through the RM port.

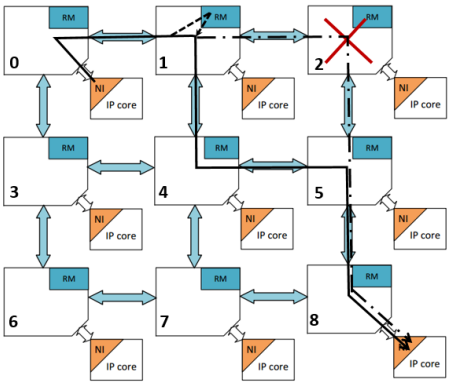

Fig. 1. Re-routing the communication path.

For instance, Fig. 1 demonstrates an example how our mechanism affects to the routing-path in Case 1 presented in [3]. Since the router number 2 is prohibited, every time a packet is transmitted from router number 0 to router number 8, the header flit of that packets must go through the RM port at the router number 1 (dash line) to be processed to change direction to solid line instead of dash-pot line. After that, all other flits of the packet will be transmitted on the solid line. After the new routing-path is updated at the source router as our proposed mechanism is applied, the following packets are transmitted on the solid line without being processed at RM port.

A. Proposed Mechanism

The proposed mechanism is composed of three phases: Tracking, Processing, and Updating as shown in Fig. 2.

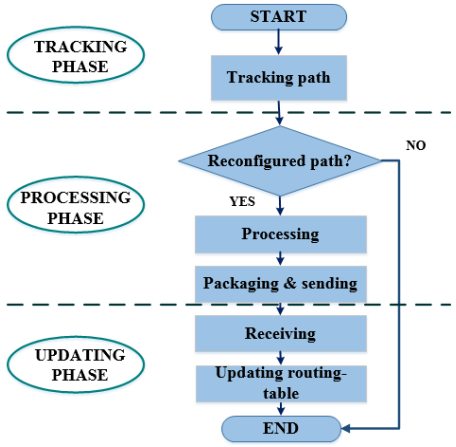

Fig. 2. The flow chart of routing-path tracking and updating mechanism.

1) Tracking phase

This phase occurs during the transmission process of every packet, and it is implemented in routers of the network. At every input port of all routers the packet is transmitted through, the packet records the previous direction it gone through. In addition, whenever packet is transmitted to RM port for reconfiguration target, the header flit of that packet will be marked. This mark is called Reconfigure mark, and it will be used in the Processing phase. Techniques used to implement these tasks will be presented in Section III.

2) Processing phase

After the Tracking phase, the tracking information (i.e. actual routing-path) will be processed at the NI of the destination router. The Reconfigure mark will be used to judge that the routing-path which packet has been transmitted through is changed or not. If that mark is detected (the value of Reconfigure mark bit is high), the tracking information will be moved to the next processing step to be formatted before being packaged and sent back to the source router. In the packaging step, the packet containing the new routing-path will be marked (Update mark). The Update mark is used in Updating phase. In the opposite case, if the Reconfigure mark is not active, the tracking information will be ignored. It means that we do not have to update the routing-path for the following packets. Then, the process is end.

3) Updating phase

The Updating phase is implemented at NI of the source router. Whenever receiving a packet, the NI will check if Update mark bit is active or not (depending on the value of Updating mark bit). If the Update mark is active (i.e. the received packet is a special packet used for updating the routing-path of the current transmission between the source and the destination), the information of this package will be used to update routing table instead of being sent to the IP core, and the process is end. If the Update mark is not active, the packet is treated as normal packet.

B. Packet format

In this work, there are two types of packets. They are defined depending on their functions as described in TABLE I. The first one is called normal packet which carries data for communication between IP-cores. The second one is a special packet containing new routing-path information (*new path2target*) used for Updating phase.

TABLE I. PACKET TYPES

Features	Normal packet	Special packet
Length (flit)	Up to IP-core requests	3
Flit size (bit)	34	
Header flit	Have non-active Update mark	Have active Update mark
Body	Contain data from IP-core	Contain new routing-path
Tail flit	Store tracking information (actual routing-path)	

The structures of each flit of each packet type are demonstrated in Fig. 3.

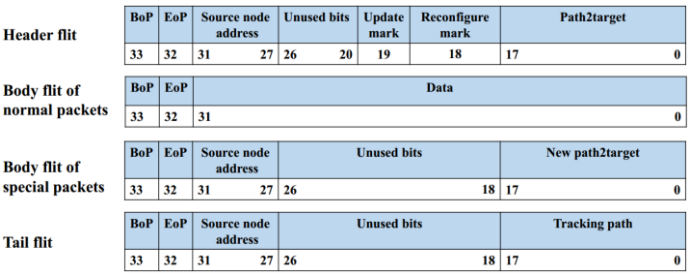

Fig. 3. Structure of flit types.

III. HARDWARE IMPLEMENTATION

To implement the proposed routing-path tracking and updating mechanism, we have modified the RNoC router in [6] and proposed a new NI architecture.

A. RNoC router modification

As the RNoC platform uses source routing algorithm, the header flits include the routing-path called *path2target*. This is presented in 18-bit field, each two bits present one direction: "*00*", "*01*", "*10*", and "*11*" for North, East, South, and West, respectively. The routing-path begins from the right of *path2target* field. The end is recognized by four consecutive bits presenting two opposite directions, "1101" for example.

Depending on the structure of *path2target*, all input ports of router except the local input port which is connected to the IP core are modified to implement the routing-path tracking function. Each input port will have a code which informs the previous direction packet was transmitted to. Codes of these ports are given in TABLE II.

TABLE II. CODE OF MODIFIED INPUT PORTS

Input port	Code	Previous Direction
North input port	10	South
East input port	11	West
South input port	00	North
West input port	01	East

In the modified RNoC, whenever a tail flit is received at one of four considered input ports, that flit will be written a code of that port into two first left bits on the *tracking path* field of the tail flit (Fig. 3). Then this field will be right-shifted 2 bits to be ready for next input port. However, the tracked routing-path stored in *tracking path* field is not in *path2target* format. Because of tracking scheme, the beginning of the tracked routing-path depends on the length of the routing-path. To support the searching of beginning of the tracked routing-path, two first left bits of *tracking path* field are default to "11" at the source router. In addition, a loop path can appear in Case 3b (see detail description in [6]) causes two opposite directions in tracked path, and this make confusion with the end of path. Therefore, in the Processing phase, the loop routing-path must be detected and deleted.

B. Network interface architecture

Block diagram of our proposed NI architecture is shown in Fig. 4. As introduced in Section II, while the Tracking phase is implemented in routers, two remain phases of our proposed mechanism are implemented in NI. Therefore, our proposed NI architecture includes additional UPDATING PATH block and some other blocks with more functions than the conventional NI architecture [7].

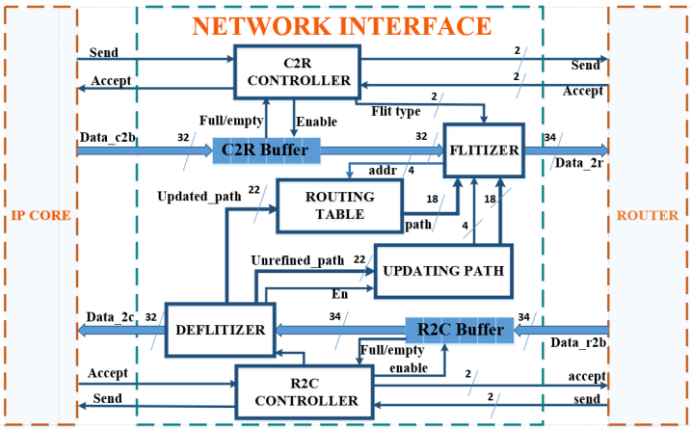

Fig. 4. Block diagram of NI.

The Processing phase is implemented in DEFLITIZER block, the UPDATING PATH block and FLITIZER block with the order corresponding to three steps in that phase shown in Fig. 2. Firstly, the Reconfigure mark is recognized by the DEFLITIZER block in de-packaging process. If the mark is detected in the header of a packet, the tail of that packet will be forward to the UPDATING PATH block through *Unrefined_path* bus.

At the UPDATING PATH block, the tracked routing-path is formatted by checking loop path and searching the beginning of the routing-path. The architecture of the UPDATING PATH block is depicted in Fig. 5. The operation this bock is described in with the finite state machine as shown in Fig. 6.

When the sytem is reset or the updating information is sent, FSM state is IDLE. Unrefined path will be loaded into 30-bit Register when the data at *unrfined_path* bus is valid (i.e. *En* = *'1'*), then that path will be formatted (FSM state is CHECKING). After the checking step (i.e. *Begin* = *'1'*), path will be sent to *Flitizer* (i.e. FSM state is SENDING).

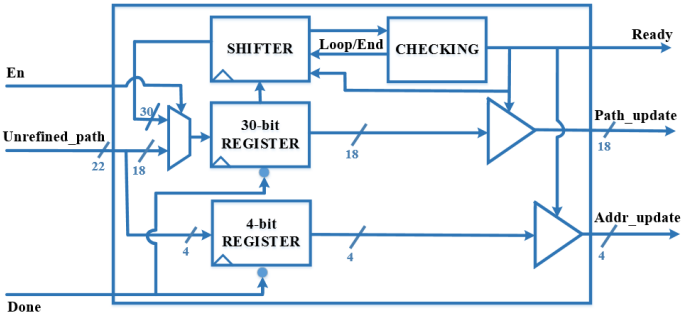

Fig. 5. Structure of UPDATING PATH block.

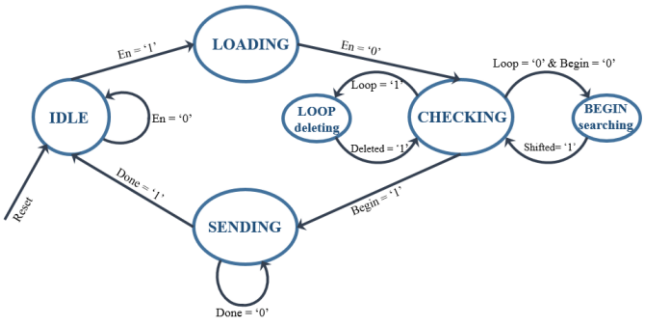

Fig. 6. FSM for the operation of UPDATING PATH block.

IV. EVALUATION

A. Methodology

To verify the operation of our proposed mechanism, we develop a testbench model as shown in Fig. 7.

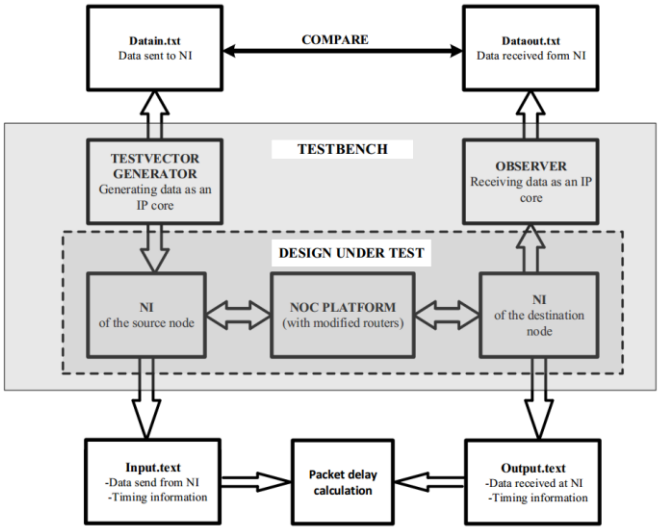

Fig. 7. Testbench model for evaluating the proposed mechanism.

This model is composed of three parts: Testvector Generator, Observer, and Design Under Test (DUT). The DUT includes a 2D 5×5 modified RNoC platform and our proposed NIs. The Testvetor Generator and the Observe emulate simple functions of IP-cores to interface with NIs.

There are two simulation scenarios. The first one is without our proposed mechanism, and the second one is with our proposed mechanism. In both scenarios, DUT is

simulated with four cases (Case 1, Case 2, Case 3a, Case 3b with loop path) of reconfiguration strategy of the work presented in [6]. The timing information about sending and receiving data at NIs (i.e, *input.text* and *output.text*) in two considered scenarios will be used to calculate the packet delay as well as the decrease in packet delay after applying our propose mechanism.

B. Results evaluation

The proposed NI is synthesized with Virtex5 XC5VLX330-2ff1760 using Xilinx tools. Obtained results are shown in TABLE III.

TABLE III. DEVICE UTILIZATION SUMARY

Logic utilization	Used	Available	Utilization
Number of Slice Registers	1334	207360	0%
Number of Slice LUTs	609	207360	0%
Number of fully used LUT-FF pairs	193	1750	11%
Number of Bonded IOBs	146	1200	12%
Number of BUFG/BUFGCTRLs	1	32	3%

In this work we defines that packet delay is the period a packet is transmitted from the NI of the source router to the NI of the destination router. The average packet delay is calculated by equation 1.

$$T_{delay} = \frac{\sum_{1}^{N} T_i}{N} = \frac{\sum_{1}^{N} (T_{i\_arr} - T_{i\_send})}{N} \quad (1)$$

where T_{delay} is packet delay; N is the number of packets; T_{i_send} is the time a header flit is sent from the source NI while T_{i_arr} is the time a tail flit is arrived at the destination NI.

The packet delay decrease is calculated by equation 2.

$$T_{decrease} = \frac{(T_{delay\_before} - T_{delay\_after}) \times 100}{T_{delay\_before}} \% \quad (2)$$

where, $T_{decrease}$ is the packet delay decrease. T_{delay_before} is T_{delay} before our proposed mechanism is applied and the T_{delay_after} is T_{delay} after applying our mechanism.

The packet delay decrease is evaluated with four cases as shown in Fig. 8. The packet delay decrease goes down with the increasing of packet size in all cases. The chart illustrates that with the same packet size, the packet delay decrease in Case 3b is the biggest while it is smallest in Case 1.

On the one hand, since our mechanism only affects directly to the header flit in all cases, the decrease in packet delay reduces with the increasing of packet size. In Case 3b with loop path, our mechanism reduces not only the time the header flit is processed at RM port, but also the time whole packet is transmitted in a loop path, so the packet delay decrease in this case is biggest.

On the other hand, packet transmission delay also depends on the routing-path length. In the simulation, the path before the appearance of prohibited router is the same in all cases. After the appearance of one prohibited router, the length of path increases in Case 1 and Case 3b while it is the same in Case 2 and Case 3b. That is one of the reasons why the packet delay decrease is the same in Case 2 and Case 3a, while it is smaller in Case 1.

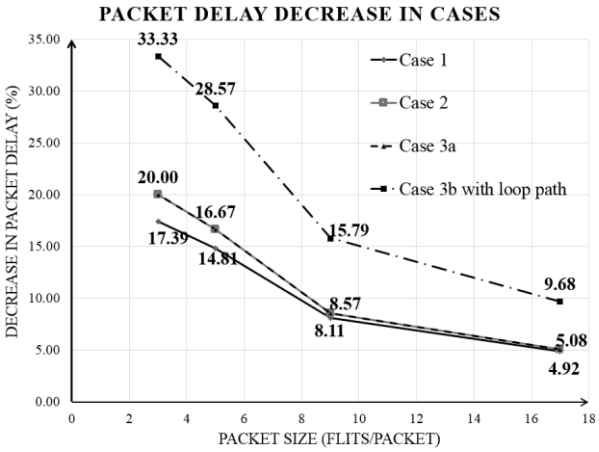

Fig. 8. Packet delay decrease in cases.

Case 3b with loop is a very different situation. After the appearance of the prohibited router, the length of routing-path is increased. However, after applying our proposed mechanism, the routing-path length is decreased since the loop path is cut. After all, the routing-path length in Case 3b is unchanged.

V. CONCLUSIONS

In this paper, a routing-path tracking and updating mechanism has been proposed. This mechanism aims at reducing the packet transmission delay in the RNoC platform. To prove the efficiency of the proposed mechanism, we have modified RNoC routers and designed NI architecture. The simulation and measurements shows that our mechanism can help RNoC to reduce from 4.92% to 33.33% in packet transmission delay, depending on the data structure.

ACKNOWLEGMENT

This work is supported by the research project No. 102.01-2013.17 (ReSoNoC) granted by Nafosted.

REFERENCES

[1] Agarwal, C. Iskander, R. Shankar. Survey of Network on Chip (NoC) architectures & contributions. Journal of engineering, Computing and Architecture, vol. 3, no. 1, pp. 21-27, 2009.

[2] R. Dafali, J. P. Diguet, M. Sevaux. Key Research Issues for Reconfigurable Network-on-Chip. International Conference on Reconfigurable Computing and FPGAs, pp. 181-186, 2008.

[3] Xuan-Tu Tran, et al. Design-for-Test Approach of an Asynchronous Network-on-Chip Architecture and its Associated Test Pattern Generation and Application. IET Journal on Computers and Digital Techniques, Volume 3, Issue 5, pp. 487-500, Sept. 2009.

[4] M. B. Stensgaard, J. Spars, and x0F, "ReNoC: A Network-on-Chip Architecture with Reconfigurable Topology." pp. 55-64, 2008.

[5] M. A. A. Faruque, T. Ebi, and J. Henkel, "Configurable links for runtime adaptive on-chip communication." pp. 256-261, 2009.

[6] L.-V. Thanh-Vu, X.-T. Tran. High-Level Modeling and Simulation of a Novel Reconfigurable Network-on-Chip Router. REV Journal on Electronics and Communications, vol. 4, pp. 68-74, 2014.

[7] Z. A. M. Adnan Mahmood, "Design and prototype of resource network interfaces for netwok on chip," Master of Science 2009.