Topology Re-Configuration for On-Chip Networks with Back-Tracking

M.Venkata Theertha
Master of Technology
Dept. of ECE
Vardhaman College of Engineering

Rajesh Nandi
Master of Technology
Dept. of ECE
Vardhaman College of Engineering

B.V.S.L.Bharathi
Asst. Professor
Dept. of ECE
Vardhaman College of Engineering

ABSTRACT
Supporting multiple applications is a critical feature of an NoC when several different applications are integrated into a single modern and complex multi-core system-on-chip or chip multiprocessor. In this paper, a novel reconfigurable architecture for networks-on-chip (NoC) on which arbitrary application-specific topologies can be implemented with backtracking which provides guaranteed throughput is presented. The proposed NoC supports multiple applications by configuring its topology to the topology which matches the input application and also supports a dead-and-live lock free dynamic path set-up scheme. The re-configurability can be achieved by changing the inter-router connections to some predefined configuration corresponding to the application. This increases the support for higher number of applications which further increases the traffic congestion leading to path blockages and substantially to data loss. To manage the blockages and to support a dead and live lock free dynamic path set-up scheme we go for back-tracking. This can be achieved with an efficient and proper design of on-chip switching nodes. This paper first introduces the proposed reconfigurable topology and then deals with the back-tracking feature. Finally the architecture is valuated for power and area.

General Terms
Network-on-Chip, Back-Tracking, Multi-SoC based Systems

Keywords
Application-specific systems-on-chip (SoCs), multi-application-based design, networks-on-chip (NoC), Back-Tracking, reconfigurable systems.

1. INTRODUCTION
Application-specific optimization is one of the most effective approaches to bridge the gap between the current and the ideal network-on-chip (NoC) power/performance metrics in application-specific multi-core systems-on-chip (SoCs) [6]. These optimization methods try to customize the architecture of a NoC for a target application, when the application and its traffic characteristics are known at design time, which is the case for most embedded applications running on multi-core SoCs.

Most NoC architectures and their design flows for application-specific multi-core SoCs provide design-time NoC optimization for a single application. In other words, they try to generate and synthesize an optimized NoC based on the traffic pattern of a single application. Recent multi-core SoCs (mainly programmable multi-core SoCs) have become highly complex and cost-effective. As technology advances, integrating several different applications into a single SoC chip becomes more cost-effective. As a result, such NoC architectures should closely match the traffic characteristics and performance requirements of different target applications. However, the inter-core communication characteristics can be very different across the applications since different applications have different functionalities. So, a NoC that is designed to run exactly one application does not necessarily meet the design constraints of other applications. Prior work shows that when conducting simulations on over 1500 different NoC configurations (with different topology, buffer size, and/or bit-width), no single NoC can be found to provide optimal performance across multiple applications.

Optimizing the network topology and core to network mapping are two important application-specific NoC customization methods which affect the network's performance and related characteristics such as average inter-core distance, total wiring length, and communication flows distribution. These characteristics determine the average network latency and power consumption of the NoC architecture. Mapping determines on which node each processing core should be physically placed while Topology determines the connectivity of the NoC nodes. Mapping algorithms generally try to place the processing cores communicating more frequently near each other; It should be noted that when the number of intermediate routers between two communicating cores is reduced, the power consumption and latency of their communications decreases proportionally.

In this paper, a NoC [9] with reconfigurable connections to dynamically change the connectivity among cores is introduced. The proposed NoC enables the network topology to dynamically match the communication pattern of the currently running application. The recon-figuration of the proposed architecture is achieved by inserting several simple switches in the network allowing the network to dynamically change the inter-node connections. Thus it implements the topology that best matches the communication pattern of the running application. Multiple topologies varying from regular tile-based[8] to fully customized structures are proposed for on-chip networks. Since fully customized NoCs are designed and optimized for some specific applications, best performance and power results are obtained for that applications. On the other hand, regular NoC architectures provide standard structured interconnects ensuring well-controlled electrical parameters. Usual physical design issues like crosstalk tolerance, timing closure, and wire routing In these topologies, can be solved for a specific regular topology and can be reused in several designs.

As the complexity of systems-on-chips (SoCs) increases, the network-on-chip (NoC) is being assumed as a communication-centric solution for integrating numerous on-

chip components i.e., processing elements (PEs), sub-blocks, and intellectual properties (IPs). Guaranteeing throughput[7] (or providing a quality-of-service (QoS) mechanism) of the traffic offered by the on-chip components, particularly with the very limited budget of the resources on the chip is one of the most difficult tasks in NoC design. Complex QoS mechanisms involved in many levels of abstraction of NoC design are required. The requirements of the performance metrics for certain traffic classes, such as data loss, data rate (throughput), delay, and delay jitter. An efficient and proper design of on-chip switching node (i.e., the switch router) is needed, while keeping compatibility. In this paper, the term "QoS property" implies the property of a switching node, such as (low) fall through latency, being lossless and ordered data delivery. This QoS Property directly supports the specified QoS requirements. Circuit switching and packet switching (at the granularity of packets or flits) are the two main themes in the literature for switching mechanisms used in the design of NoC routers. These NoC routers impact greatly on the implementation overhead and the QoS property.

The proposed NoC architecture benefits from both worlds and can be placed between these two extreme limits of NoC design schemes. Even though this NoC architecture is designed and optimized like regular NoCs, it can be reconfigured dynamically to a topology that best matches the traffic pattern of the currently running application. In other words, application-specific topologies over structured and regular components are realized by these architectures. The main contributions of this work are as follows.

1) First, we propose a novel reconfigurable NoC architecture on which a range of regular and application-specific topologies[7] can be implemented.

2) Second, we propose the back-tracking feature which manages the blockages and supports a dead and live lock free dynamic path set-up scheme by appropriately configuring the NoC connections. This can be achieved with an efficient and proper design of on-chip switching nodes.

The rest of this paper is organized as follows. In Section 2, presents the proposed re-configurable NoC architecture followed by Section 3 which deals with the Back-Tracking feature for the proposed reconfigurable NoC. The experimental results appear in Section 4 where the performance and power consumption of the proposed NoC architecture is evaluated. In Section 5, simulation results are reported. Finally, concluding remarks are given in Section 6.

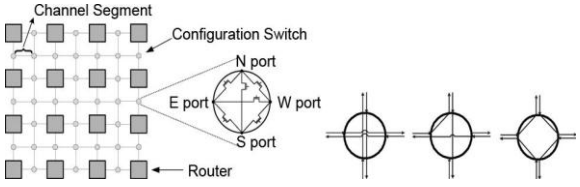


Fig. 1. (a) Reconfigurable NoC architecture (b) three possible switch configurations.

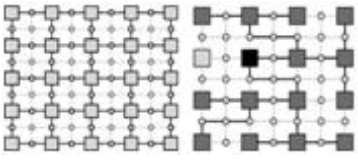


Fig. 2. Reconfigurable NoC configured as (a) a mesh and (b) a binary tree.

2. PROPOSED NOC ARCHITECTURE:

2.1 Proposed NoC Architecture:

The architecture considered is composed of $m \times n$ nodes arranged as a 2-D mesh network. However, the routers are not connected directly to each other in the proposed NoC architecture, they are connected through simple switch boxes called configuration switches (see Fig. 1). Each square box in Fig. 1 represents a network node which is composed of a processing element and a router. Each circle represents a configuration switch which is the major component that differs from regular Nocs. Fig. 1(a) also shows the internal structure of a configuration switch. It consists of some simple transistor switches. These switches can establish connections between incoming and outgoing links. In this figure, only a single connection is depicted between each two ports of a configuration switch.

In order to route the incoming and outgoing sub-links of bidirectional links independently, there are two connections between each two ports of a configuration switch. For example, the outgoing sub-link of the N (north) port can be connected to the incoming sub-link of some port (the S port, for example) while the outgoing sub-link of the N port is connected to the incoming sub-link of a different port (the W port, for example). At each output port of the switch the internal connections are implemented by using a multiplexer. Each multiplexer is connected to three input ports, since a connection coming through an input port does not loop back. Fig. 1(b) displays three possible switch configurations.

If the configuration switches are set properly this NoC can be configured based on arbitrary topologies, including some standard topologies. For example, Fig. 2 display a 5 X5 network configured as a 2-D mesh [see Fig. 2(a)] and a 4 X 4 network configured as a binary-tree [see Fig. 2(b)].

In general, reconfigurable devices can be used for dynamic hardware reconfiguration; hence, Most of the reconfigurable architectures are implemented using field-programmable gate arrays (FPGAs). This NoC can be implemented on both FPGA and application-specific integrated circuit (ASIC) platforms, since the Re-configurable part of the proposed NoC is limited to some simple configuration switches.

The proposed reconfiguration mechanism can be applied to well-known topologies, such as torus, hypercube, and k-ary n -cubes. Here the mesh topology is used as the basis of the reconfigurable NoC architecture. It can also be used in some modern topologies, such as concentrated mesh (or X-mesh) and WK-recursive networks, to further improve their performance when dealing with multiple applications.

2.1 Cost of the Proposed Architecture:

Like FPGAs, the proposed reconfigurable network pays for the flexibility with additional area overhead. In this section, we evaluate the area of the proposed NoC architecture and compare it to a conventional packet-switched NoC. Same structure is considered for a reconfigurable NoC, but it includes some additional area due to the configuration switches. The area estimation is done based on the area model employed in RC Compiler.

A typical NoC is composed of three types of components: routers, network interfaces (NI), and links. Our proposed NoC uses an extra component, i.e., the configuration switch which is responsible for reconfiguration. Based on the area of the input buffers router area is estimated. The router area is

dominated by these two components. Network interfaces generally include packetization and de-packetization logics. To implement internal connections each configuration switch is composed of four 3 × 1 multiplexers. The area of a configuration switch is much less than the area of a router since it has small buffering capacity, no controller logic (for virtual channel and switch allocator, and routing logic), uses a smaller switching fabric and no network adapter.

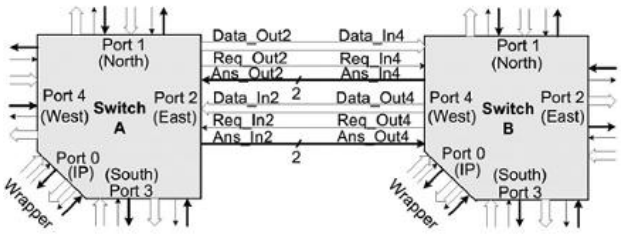
By calculating the number of channels in a conventional and a reconfigurable mesh-based NoC, the area overhead due to the additional inter-router wires is analyzed. We consider two channel-segments (see Fig. 1) in the reconfigurable NoC (which has the same size as a channel in a conventional NoC) as one channel. Square-shaped cores of 1 mm × 1 mm are assumed. Consequently, each NoC channel is 1 mm long. The channel area model is calculated using NC Sim.

The storage space required to keep the configuration information of the switches is another source of overhead. The configuration data for several applications can be kept in the switch with moderately low storage requirements since each configuration switch is configured by eight bits divided into four two-bit parts each to control one of the internal multiplexers of the switch.

In this work, whenever a switching takes place between two applications it is assumed that the NoC reconfiguration process is initiated by a configuration manager (which may be implemented in the application layer). In addition to the area overhead discussed above, dynamic reconfiguration of the NoC may generally cause some performance overhead due to the time needed to load a new configuration to the NoC. However, in most SoC designs, the application switching time is of the order of few milliseconds. This switching time consists of the time needed to load the data and code of a new application into the SoC, shutting down the old application [3] and sending control signals to different parts of the SoC. Switching between network configurations is done in parallel with application switching. The configuration data of the proposed NoC architecture is small and can be stored in configuration switches and routers. This reduces the NoC configuration-switching time and this is by far smaller than the time needed to switch between applications. The main advantage is that this does not impose any additional delay to the application switching procedure. It has been shown in [3] that even if the configuration data is stored in an off-chip memory, it can be loaded and distributed around the NoC in few microseconds which is still shorter than the application switching time. Due to infrequent switching and the small amount of data transition during each switching event, the energy dissipated for configuration switching can also be ignored.

3. BACK-TRACKING SWITCH ARCHITECTURE:

Back-tracking switch architecture [7] for the proposed NoC is achieved with an efficient and proper design of on-chip switching nodes. When congestion occurs, instead of waiting for a busy link to become idle the probe header needs to backtrack and search for alternative links. The path is set up under a backtracking probing path-setup scheme. An ACK signal is returned to the source when the probe header reaches its destination. In the transmission phase, the source starts to transmit source-synchronous data through the set up path to the destination. A compact switch-by-switch handshake is proposed to support such end-to-end communication with the probing path-setup scheme. Fig. 3 illustrates the inter-switch and switch-wrapper interconnections with this handshake. There are five bidirectional ports in each switch: four ports are connected to corresponding neighboring switches, and the remaining port is connected to the on-chip IP through a wrapper.



EXPERIMENTAL RESULTS:

To evaluate the performance of the proposed NoC architecture, simulations for some benchmark applications have been performed. VOPD and MWD applications have been considered for the simulations, whose topologies have been taken from [5]. The applications use the same set of IP-Cores but the traffic pattern among the cores is different for each different application.

We have evaluated the proposed NoC architecture using NC Sim. The power results reported are based on NoC implemented in 90 nm technology and the working frequency of the NoC is set at 200 Mhz for both the applications. The power and area results of the proposed architecture are compared with the ReNoC architecture proposed in [2].

Table 1. Comparison of the proposed ReNoC and ReNoC for VOPD

	Proposed ReNoC	ReNoC	Improvement Over
Power	0.119	0.107	10%
Normalized area	1	0.88	12%

The results show that the area of a 4×4 ReNoC is 12% less than the area of the proposed reconfigurable NoC of the same size. Simulation results reported in Table 1, show that the proposed architecture consumes very less extra power and area overhead compared to the ReNoC, even though it provides both re-configurability and Back-tracking. In this situation, the power of the conventional NoC does not increase due to blocked packets, while its reconfigurable counterpart is still working normally, and transferring more packets which in turn consume more power.

We have also compared our proposal with ReNoC, the reconfigurable NoC architecture proposed in [2]. As ReNoC considers no specific algorithm for topology generation, we consider the VOPD application for which the topology is provided [2] on a 4 × 3 mesh. We assume that the packets are sent over the long links in a pipelined fashion. We use the same simulation parameters used earlier in this section for the

VOPD application. Area model shows that the area of a 64-bit 4 × 3 ReNoC is 12% less than the area of our proposed reconfigurable NoC of the same size.

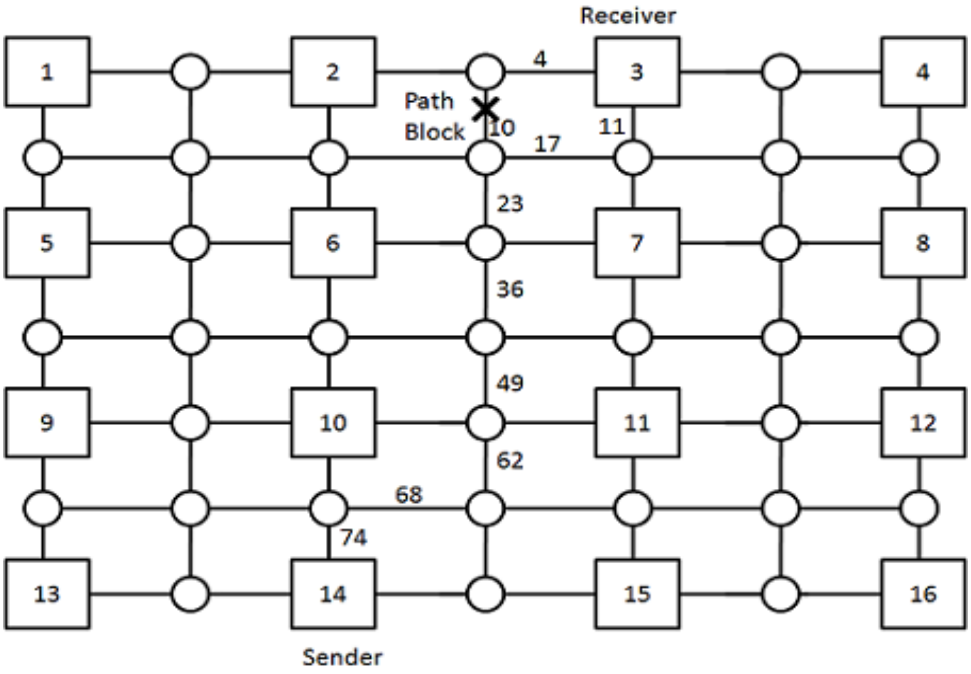


Fig 4. A 4×4 NoC showing Blockage on the path between the Sender and the Receiver

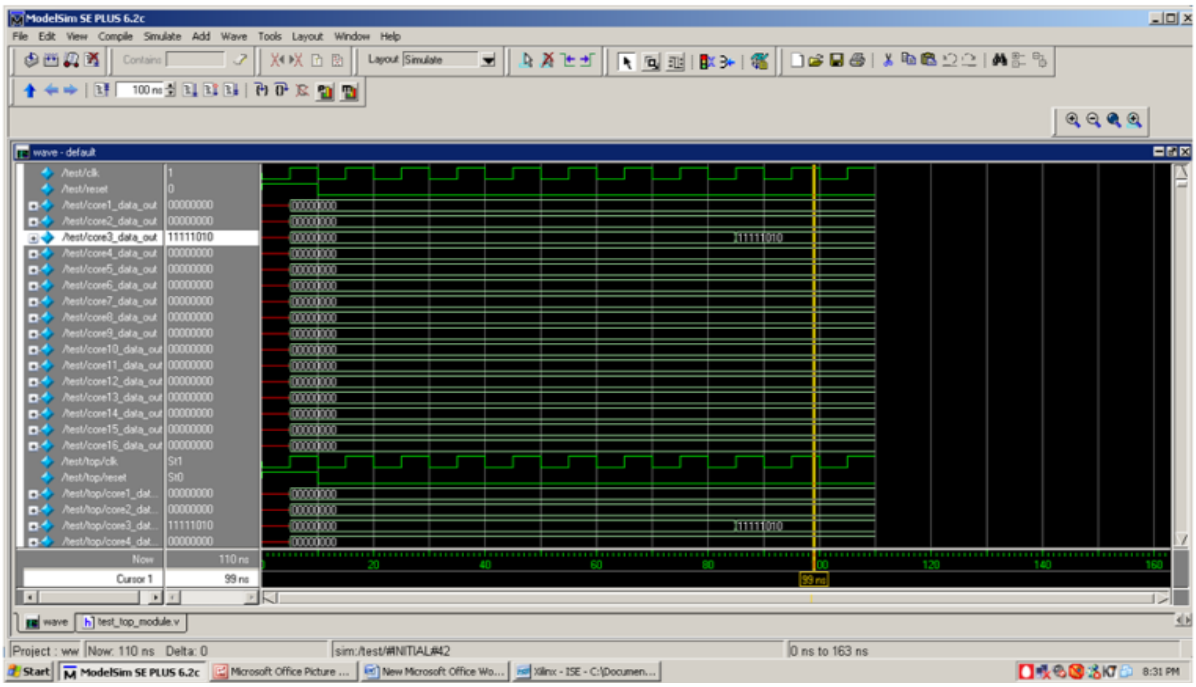


Fig 5. Synthesis result of the proposed NoC without blockage.

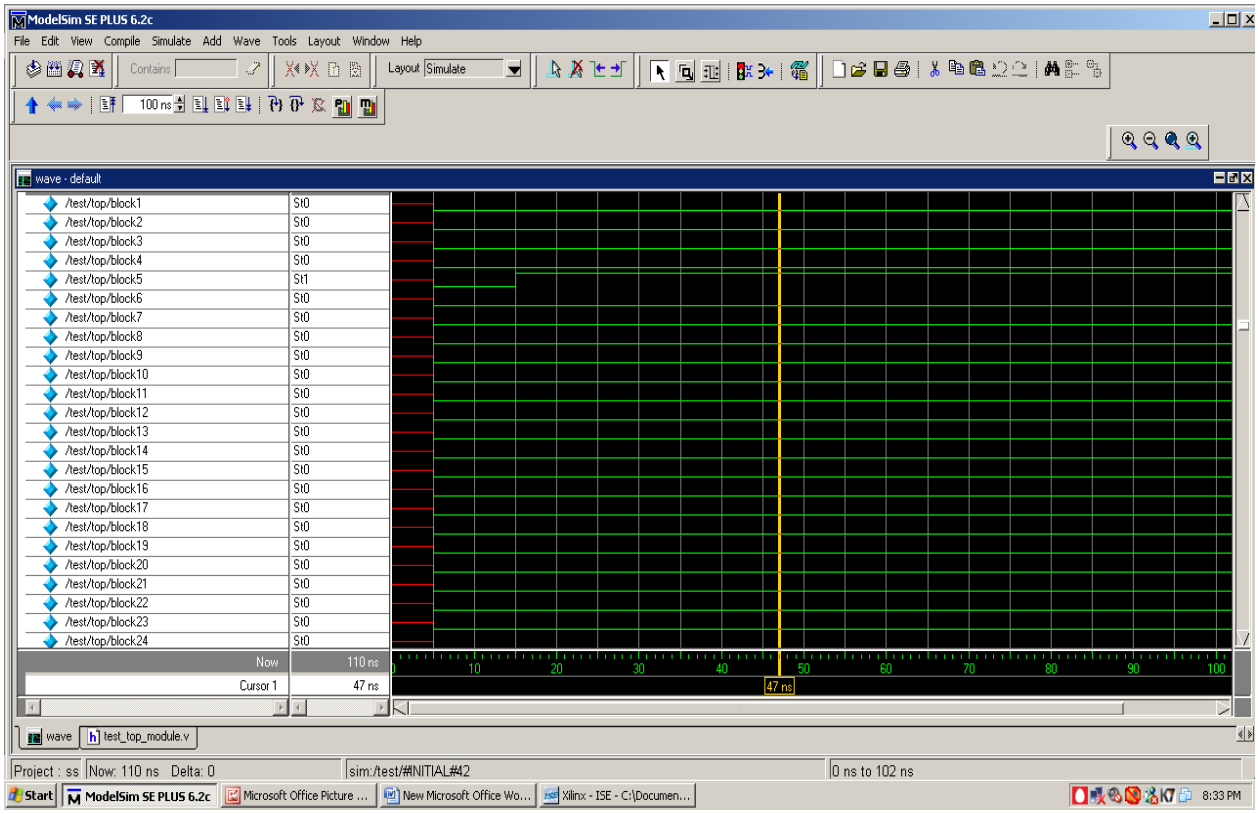


Fig 6. Synthesis result showing Path Blockage

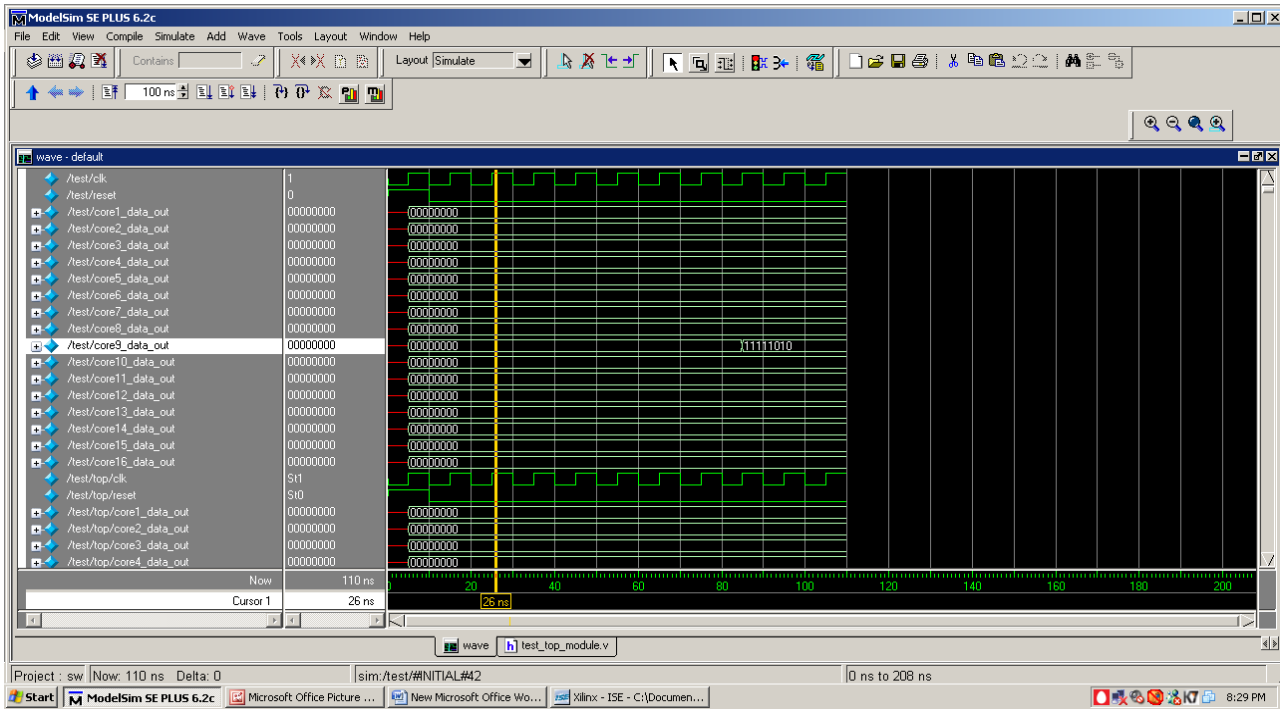


Fig 7. Synthesis result of the proposed NoC With Back-Tracking feature implemented

For simulation, a 4×4 NoC on which VOPD application is mapped is considered and the cores are numbered from 1 to 16. Similarly nets are numbered from 1 to 84 which is shown in the fig 4. It is assumed that the Sender is the Core 14 and receiver is the Core 3. Suppose if for a particular task the data has to be sent along the path containing the nets, 74 > 68 > 62 > 49 > 36 > 23 > 10 > 4. To illustrate the concept of Back-Tracking a path blockage is assumed on the net numbered 10. The blockage could be due to physical damage of the net or traffic congestion etc. As discussed earlier the Re-configurable switches are properly and efficiently designed so that the blockage is handled by sending the data through an alternate route which is pre-configured within the switch. As pre-configured the switching node at node 7, resends the data in the path 74 > 68 > 62 > 49 > 36 > 23 > 17 > 11, thus delivering the data to the receiver and making the circuit a live-lock free circuit.

5. CONCLUSION:

A reconfigurable architecture for a NoC on which arbitrary application-specific topologies can be implemented has been proposed. Since entirely different applications may be executed on an SoC at different times, the on-chip traffic characteristics can vary significantly across different applications. However, almost all existing NoC design flows and the corresponding application-specific optimization methods customize NoCs based on the traffic characteristics of a single application. The re-configurability of the proposed NoC architecture allows it to dynamically tailor its topology to the traffic pattern of different applications.

In this paper, first a reconfigurable NoC architecture is implemented and its implementation cost in terms of area overhead over a conventional NoC is evaluated. Then a special feature which provides dead and live lock free circuits for High Traffic ReNoCs called Back-Tracking has been introduced. Experimental results, using some multi-core SoC benchmarks, showed that this architecture effectively improves the performance of NoCs by 29% and there is only slight improvement in the power consumption which is 7%, over one of the most efficient and popular mapping algorithms proposed for conventional NoCs. Compared to previous reconfigurable proposals [2] and regarding the imposed area overhead and power/performance gains, the proposed NoC introduces a more appropriate tradeoff between the area and flexibility.

6. REFERENCES

[1]. M. Modarressi and H. Sarbazi-Azad, "Application-Aware Topology Reconfiguration for On-Chip Networks," IEEE Transactions on,Very Large Scale Integration (VLSI) Systems, 2011, pages : 2010-2022

[2]. M. Stensgaard and J. Sparso, "ReNoC: A network-on-chip architecture with reconfigurable topology," in *Proc. Int. Symp. Networks-on-Chip (NoCS)*, 2008, pp. 55–64.

[3]. M. Modarressi, "A reconfigurable topology for NoCs," Tech. Rep. TR-HPCAN10-2, 2010.

[4]. Jiajia Jiao; Yuzhuo Fu; "Multi-application Specified Link Removal Strategy for Network on Chip," Computational Sciences and Optimization (CSO)," Fourth International Joint Conference, 2011.

[5]. M. Modarressi, H. Sarbazi-Azad, and A. Tavakkol, "An efficient dy-namically reconfigurable on-chip network architecture," in *Proc. Des. Autom. Conf. (DAC)*, 2010, pp. 310–313.

[6]. J. Owens, W. J. Dally, R. Ho, D. N. Jayasimha, S. W. Keckler, and L. S. Peh, "Research challenges for on-chip interconnection networks," *IEEE Micro*, vol. 27, no. 5, pp. 96–108, May 2007.

[7]. Design and Implementation of Backtracking Wave-Pipeline Switch to Support Guaranteed Throughput in Network-on-Chip, **by** Phi-Hung Pham et al. S. Murali, M. Coenen, R. Radulescu, K. Goossens, and G. De Micheli,

[8]. "A methodology for mapping multiple use-cases onto networks on chips," in Proc. Des. Autom. Test Euro. (DATE), 2006, pp. 118–123.

[9]. S. Murali and G. De Micheli, "Bandwidth-constrained mapping of cores onto NoC architectures," in Proc. Des. Autom. Test Euro. (DATE), 2004, pp. 896–901.

[10]. L. Benini and G. De Micheli, "Networks on chip: A new paradigm for systems on chip design," IEEE Comput., vol. 35, no. 1, pp. 70–78, Jan. 2001.