PNOC: Implementation on Verilog for FPGA

Uzma Mushtaq and Osman Hasan
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan
{09mseeumushtaq, osman.hasan}@seecs.nust.edu.pk

Falah Awwad
Faculty of Information Technology,
UAE University,
Al-Ain, UAE
f_awwad@uaeu.ac.ae

***Abstract*—Network on Chip (NoC) architectures provide a very efficient means for performance enhancement in digital circuits. The paper describes a NoC implementation that is specifically targeted towards FPGA based designs. Our implementation is based on a lightweight circuit-switched architecture called programmable NoC (PNoC). It is captured in the Verilog hardware description language and is implemented using the Xilinx Virtex-II pro FPGA (XC2Vp30-7) device at 126 MHz. The proposed architecture allows parametrization at the compile time for the number of nodes and amount of data. Moreover, experimental results have confirmed that the proposed implementation is the most efficient one in terms of performance.**

I. INTRODUCTION

The high nonrecurring engineering (NRE) costs and long time to market for ASICs are making FPGAs more appropriate to be used in different hardware applications [1]. The current FPGAs consist of a large amount of Configurable Logic Blocks (CLBs), which may be used to resolve the small memory limitations of traditional FPGAs. However, efficient hardware implementations have to be sought in order to exploit the full potential of FPGA based designs. In this respect, Network on Chip (NoC) [2] provides one of the most efficient hardware implementations on the FPGA platform.

NoC is basically a computing mechanism that consists of several interconnected processors. It provides many benefits for design exploration and performance analysis by the usage of IP macro-cells in a plug and play fashion. Scalability, design-flow parallelization, and reusability are the main benefits of NoC based implementations and they greatly help in reducing time-to-market and development costs. However, transferring a traditional hardware design to a NoC based design is not a very straightforward task as it involves partitioning of the design in processing units and defining their communication mechanism.

In order to facilitate the process of porting a traditional hardware implementation to a NoC based implementation, a simple circuit-switched architecture called programmable NoC (PNoC) was proposed in [3]. PNoC is a very flexible and lightweight architecture for FPGA based systems. It uses a modular design that facilities the usage of standard interfaces and IPs [4]. Higher communication bandwidth and better scalability are the foremost merits of PNoC. However, the architecture that was developed in [3] was given in JHDL [5], which is not a very commonly used hardware description language (HDL). In this paper, we overcome this problem by presenting a Verilog (a more commonly used HDL) implementation of the PNoC architecture. Besides the implementation details, the paper also presents a detailed analysis of our model and the results have been found to be consistent with the JHDL based implementation proposed in [3].

The rest of the paper is organized as follows: Section II provides some preliminary information regarding the NoC architectures. We present an overview of the related work in Section III. Section IV describes our Verilog implementation of the PNoC architecture. The results and comparisons with the PNoC architecture, proposed in [3], are presented in Section V. Finally, Section VI concludes the paper.

II. PRELIMINARIES

In this section, we provide some fundamentals regarding the NoC architectures. These fundamentals would facilitate the understanding of the rest of this paper.

As the system complexity increases, the bus based communication systems are not efficient in terms of scalability, and design and verification times. Moreover, the demand for higher computational speeds and lower power consumption is essentially required in most of the consumer electronics and medical equipment. The NoC technology provides a possible solution to all of these problems.

The basic ingredients of a NoC architecture, depicted in Fig. 1, include the processing elements, connection topology, routing technique, switches, and programming model. There are various connection topologies from the communication perspective. Torus, octagon, mesh, ring, and irregular connection networks are some of the communication topologies [6, 7]. Several researches have shown that the 2-D mesh architecture is easy to implement and provides the most efficient solution for low latency [8].

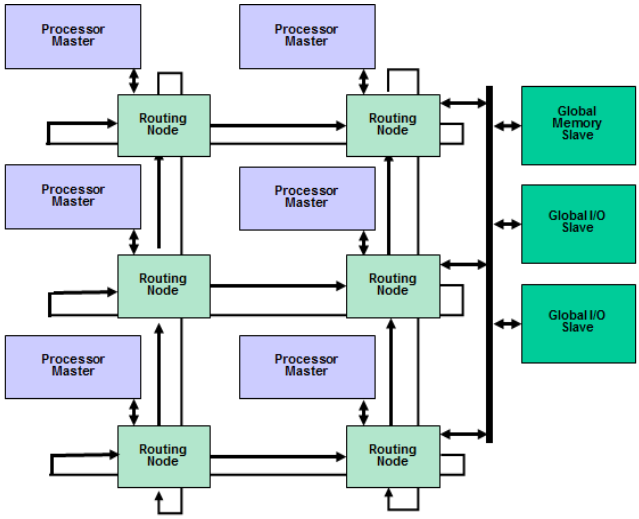

Fig. 1. Example of NoC.

Different architectures for routing have been proposed but the most widely used ones are packet switched and circuit switched. The packet switched routing architecture [9] requires some amount of buffering. Buffers are provided at input, output or both input and output. In the circuit switched architecture [10], there is a dedicated channel for the data flow so no buffering or queuing is required.

Switching techniques are classified according to the network as illustrated in Fig. 2. Circuit switching networks reserve the path before transmission of data while packet switched networks use buffers to store data. Three switching techniques are commonly used in the case of packed switched networks: Virtual cut through algorithm divides packets into flits and then into phits. This switching technique has not been adopted by any of the NoC algorithms. In store and forward (S&F) switching, the packet is routed only if the receiving buffer has enough space to accommodate this packet [11]. In the Wormhole switching [12], latency is experienced only by the header whereas the remaining packet follows the header.

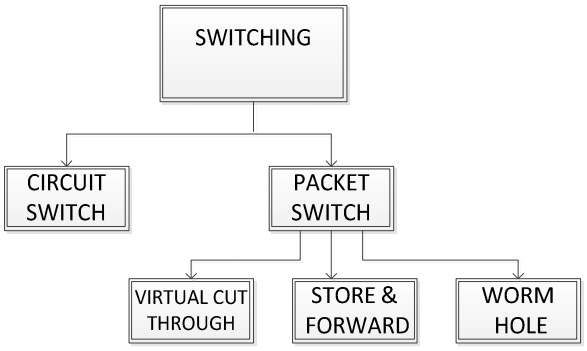

Fig. 2. Switching Techniques.

III. Related Work

The NoC technology tends to provide high performance and low power consumption for system-on-chip architectures. NoCs have been implemented in different configurations and provide various capabilities. This section provides a summary of some of the existing NoC architectures.

Xpipes architecture [13] is based on a heterogeneous model unlike NoCs, which are primarily based on a homogeneous model. Due to its heterogeneous nature, Xpipes requires an enormous amount of effort during the implementation phase. Agarwal and Shankar exploited the high performance computing capability of NoCs and proposed a layered architecture for the NoC based systems [1]. Their architecture constitutes different domains, including protocol on a NoC environment, algorithms, applications, and RTOS. Their proposed design was parameterized. This implementation uses Mission Level Designer (MLD), which is not a commonly used environment and thus is not suitable for our development where we want to provide a generic NoC implementation. One of the commercially available NoC is Arteris [14]. However, using Arteris involves hefty licensing fee due to its commercial nature. The virtual channel (VC) is an important ingredient of NoC. In [15], Bjerregaard and Sparsø proposed a new implementation of a virtual channel router and used the asynchronous techniques for implementation of the circuit. Mapping hardware onto this kind of an asynchronous circuit involves significant amount of design efforts. Al-Tawil et al. provided a survey of different switching techniques with main emphasis on wormhole routing technique [12]. The CLICHÉ NoC implementation uses store-and-forward (S&F) switching [12]. This type of switching technique uses a huge amount of buffer space available at each node. Kumar et al. implemented Nostrum NoC using S&F switching [16]. Various switching techniques can be combined to develop Ad-hoc switching techniques [17, 18]. Siebenborn et al. used Communication Dependency Graph (CDG) for inter-process communications [19]. For using the CDG, we have to perform load balancing, placement, and fanout routing, which is quite cumbersome. Neeb et al. compared torus, mesh, and cube networks [20].

In this paper, our focus is to use a flexible and lightweight NoC architecture. Most of the above mentioned NoC architectures, are either too complex to be used for this specific purpose or are not very generic. To fulfill our requirements, we identified a lightweight circuit-switched NoC for FPGA based systems, described by Hilton and Nelson in [3]. This architecture, called programmable NoC or the PNoC, is designed having three goals in mind. Firstly, the approach is flexible. Secondly, it is a simple architecture that uses standard network interfaces and simplified network protocols. Finally, it offers a lightweight network that requires a few FPGA resources, and is suitable for smaller as well as larger FPGA-based systems. However, the architecture has been described using JHDL, which is not a commonly used HDL. In order to facilitate broad usage of the

PNoC proposed in [3], we ported its design to Verilog in the current paper.

IV. PNOC ARCHITECTURE

PNoC is a circuit switched technique that simplifies system design by providing flexible networking approach and is efficient in design and verification methodologies. The network consists of subnets, in which each subnet has a router and a bunch of network nodes shown in Fig. 3 [3].

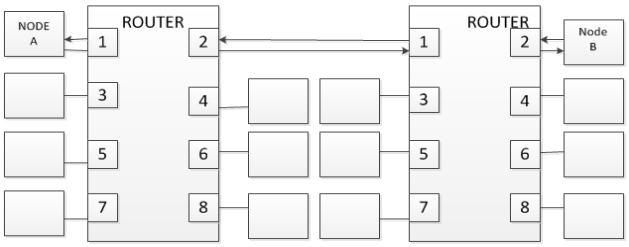

Fig 3. PNoC Network Topology [3].

The circuit switching between the nodes is performed by the router and each node is connected to a router by the router-port interface. A dedicated connection is established using light handshaking mechanism for the data exchange and connection removal.

The connection is established when master node (A) sends the request signal along with the address of the target node (B) to the router. The second router sends the grant signal to the first router that port B is available and the connection is established.

A dedicated connection path is used for data transfer so no acknowledge signal is required. Data transaction can occur on successive clock cycles if master receive is low. The read and write requests can be pipelined.

A CPU is connected to the PNoC like any other module. The interfacing circuit constitutes FIFO's and FSM to communicate with the router.

The router is the main component of the PNoC. The router includes the routing table, queue, and switch box. Another part of the PNoC is the buffer which has a parameterizable feature. Buffer is necessary in two cases. Firstly, if nodes and routers are running at different clock rates, and secondly, when there is a difference between the transmitting and receiving rate.

Our Verilog code consists of a main or top level Router module and three sub-modules and its overall structure is illustrated in Fig. 4. The Buffer module performs two main functionalities, i.e., FIFO control and the memory. This FIFO control monitors different signals for the buffers like the buffer empty, buffer full, memory pointer etc, whereas, the memory is used to store the data that comes into the buffer.

The MUX module is basically used for the selection of data lines on the basis of its select lines. The select lines of the MUX correspond to the address lines of the memory. The Route module is the switching module for the implementation. It directs the data to a path, which is decided on the basis of data and address coming from the Buffer and the MUX. The Router is the main top level module which connects all the lower level modules.

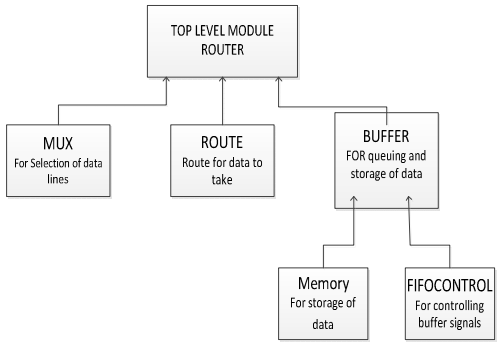

Fig. 4. Verilog Modules for PNoC

V. EXPERIMENTAL RESULTS

Table 1 shows the comparison of Hilton and Nelson's [3] and the proposed NoC implementations. The setup was designed to exhibit maximum data transfer ability. The same device and port and data widths were used. We got similar results for both implementations in spite of different languages used for implementing the architectures.

S. no	Specification	Hilton and Nelson's PNoC	Proposed PNoC
1	Device	Xilinx Virtex-II Pro FPGA (xcv2p30-7)	Xilinx Virtex-II Pro FPGA (xc2vp30-7)
2	Language	JHDL	Verilog
3	Clock Speed	126 MHz	126.486 MHz
4	Ports	8	8
5	Data Width	8	8
6	Area(Slices)	1223	1227

Table 1: Comparison of different NoC architectures

Table 2 shows the improvement of circuit switched PNoC over traditional hardware implementations.

S. no	Specification	Hilton and Nelson's PNoC	Proposed PNoC	Shared Bus	Locked Bus	Packet Switch Net.
1	Device	Xilinx Virtex-II Pro FPGA (xcv2p30-7)	Xilinx Virtex-II Pro FPGA (xc2vp30-7)	Xilinx Virtex-II Pro FPGA (xc2vp30-7)	Xilinx Virtex-II Pro FPGA (xc2vp30-7)	Xilinx Virtex-II Pro FPGA (xc2vp30-7)
2	Language	JHDL	Verilog	JHDL	JHDL	JHDL
3	Clock Speed	126 MHz	126.486 MHz	108 MHz	98 MHz	50 MHz
4	Ports	8	8	8	8	8
5	Data Width	8	8	8	8	8
6	Area(Slices)	1223	1227	689	778	2400

Table 2: Comparison of PNoC architectures with Traditional Methods (The figures excluding the proposed PNoC Scheme has been obtained from [3])

VI. CONCLUSIONS

The paper describes a NoC implementation targeted towards the FPGA technology. For this purpose, we used PNoC, which is a very flexible architecture and thus specifically suits the FPGA-based systems. Our design description is captured in the Verilog HDL and is implemented on Virtex-II pro FPGA (XC2Vp30-7) device at 126 MHz. The proposed design is parameterizable at the compile time for the number of nodes and amount of data. The experimental results have shown a significant improvement in performance when compared with the other hardware implementations of the PNoC.

REFERENCES

[1] A.Agarwal and R. Shankar, "A layered architecture for NOC design methodology", IASTED International Conference on Parallel and Distributed Computing and Systems, pp. 659-666, 2005.

[2] L. Benini and G. De Micheli, " Networks on Chip: a New SoC Paradigm", IEEE Computer, volume1, pp. 70–78, 2002.

[3] C.Hilton and B. Nelson, "PNoC: A flexible circuit-switched NoC for FPGA-based systems," IEE proceedings computers and digital techniques, volume 153 Issue 3, 2006.

[4] E. Salminen, V. Lahtinen, K. Kuusilinna, and T. Hamalainen, "Overview of bus-based system-on-chip interconnections," in Proceedings of the IEEE International Symposium on Circuits and Systems. ISCAS 02, , pp. 372–375 vol.2, 2002.

[5] Hutchings, B., Bellows, P., Hawkins, J., Hemmert, S., Nelson, B., and Rytting, M., "A CAD suite for high-performance FPGA design", in Pocek, K.L., and Arnold, J.M. (Eds.). Proc. IEEE Workshop on FPGAs for Custom Computing Machines, Napa, CA, USA, pp. 175-184, 1999, (IEEE Computer Society).

[6] J. Dally and B. Towles Principles and Practices of Interconnection Networks, Morgan Kaufmann, 2004.

[7] A. Ivanov, C. Grecu, M. Jones, P. Pratim Pande, and R. Saleh, P. Pratim Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Performance evaluation and design trade-offs for network-on-chip interconnect architectures",IEEE Transactions on Computers, volume54, no. 8, pp. 1025-1040, 2005.

[8] Hosseinabady, M.; Kakoee, M.R.; Mathew, J.; Pradhan, D.K.; , "Low Latency and Energy Efficient Scalable Architecture for Massive NoCs Using Generalized de Bruijn Graph," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.19, no.8, pp.1469-1480, Aug. 2011.

[9] J. W. Dally and B. Towles, "Route packets, not wires:"On-Chip interconnection networks", Proc. IEEE International Conference on Design and Automation, pp. 684-689,2001.

[10] P. T. Wolkotte, G. J. M. Smit, G. K. Rauwerda, and L. T. Smit, "An energy-efficient reconfigurable circuitswitched network-on-chip", Proc. 19th IEEE International Conference on Parallel and Distributed Processing Symposium, pp. 155-163, 2005.

[11] M. Millberg, E. Nilsson, R. Thid, S. Kumar, and A. Jantsch. "The Nostrum backbone - A communication protocol stack for networks on chip", Proc. IEEE International Conference on VLSI Design, pp. 693, 2004.

[12] K. M. Al-Tawil, M. Abd-El-Barr, and F. Ashraf, "A survey and comparison of wormhole routing techniques in mesh networks", IEEE Network, volume11, pp. 38–45, 1997.

[13] D. Bertozzi and L. Benini, "Xpipes: A network-on chip architecture for gigascale systems-on-chip", IEEE Circuits and Systems Magazine, volume4, Issue 2, pp. 18-31, 2004.

[14] ARTERIS. A comparison of network-on-chip and buses.Whitepaper,2005. http://www.arteris.com/nocwhitepaper.pdf.

[15] T. Bjerregaard and J. Sparsø, "A router architecture for connection-oriented service guarantees in the MANGO clockless Network-on-Chip", Proc. Of IEEE on Design Automation and Test, volume2, pp. 1226-1231, 2005.

[16] S. Kumar, A. Jantsch, J-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, "A network on chip architecture and design methodology", IEEE Computer, pp. 117-124, 2002.

[17] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, and E. Waterlander, "Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip", IEE Proc. on Computers and Digital Techniques, volume150, Issue 5, pp. 294-302, 2003.

[18] K. Goossens, J. Dielissen, and A. Radulescu, "A Ethereal network on chip: Concepts, architectures, and implementations", IEEE Design & Test of Computers, volume22, Issue 5, pp. 414-421,2005.

[19] A. Siebenborn, O. Bringmann, and W. Rosenstiel, "Communication analysis for network-on-chip design", Proc. IEEE International conference on Parallel Computing in Electrical Engineering, pp. 315-320, 2004.

[20] C. Neeb, M. Thul, and N. Andwehn, "Network on-chip-centric approach to interleaving in high throughput channel decoders", Proc. IEEE International Symposium on Circuits and Systems, pp. 1766–1769, 2005.