International Journal of Electronics
and Communication Engineering (IJECE)
ISSN(P): 2278-9901; ISSN(E): 2278-991X
Vol. 3, Issue 6, Nov 2014, 1-8

International Academy of Science,
Engineering and Technology
Connecting Researchers; Nurturing Innovations

DESIGN AND IMPLEMENTATION OF AN ON-CHIP PERMUTATION NETWORK FOR MULTIPROCESSOR SYSTEM-ON-CHIP

A CHENGAIAH[1] & Y MURALI MOHAN BABU[2]

[1]Student, Department of Electronics and Communication Engineering, Sri Venkateswara College of Engineering & Technology (Autonomous), Chittoor, A.P, India

[2]Professor, Department of Electronics and Communication Engineering, Sri Venkateswara College of Engineering & Technology (Autonomous), Chittoor, A.P, India

ABSTRACT

This paper presents the silicon-proven design of a novel On-chip interconnection networks or Network-on Chips (NoCs) are becoming the de-facto scaling communication techniques in Multi-Processor System-on-Chip (MPSoC) or Chip Multiprocessor (CMP) environment. The proposed network design configured with a 25-bit data width is synthesized and implemented in a 0.13-µm CMOS STD-cell technology. 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. A test chip comprised of 25 testing tiles is designed to test the network. Each testing tile has a 50-bit RISC and FIFO-based test wrappers interfaced with the proposed on-chip network. Tools required Modelsim 6.3 for Debugging and Xilinx 14.3 for Synthesis and Hard Ware Implementation.

KEYWORDS: Guaranteed Throughput, Multistage Interconnection Network, Network-On-Chip, Permutation Network, Pipelined Circuit-Switching, Traffic Permutation

I. 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, 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. Standard permutations of traffic occur in general-purpose MPSoCs, for example, polynomial, sorting, and fast Fourier transform (FFT) computations cause shuffled permutation, whereas matrix transposes or corner-turn operations exhibit transpose permutation. Recently, application-specific MPSoCs targeting flexible Turbo/LDPC decoding have been developed, and they exhibit arbitrary and concurrent traffic permutations due to multi-mode and multi-standard feature. In addition, 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.

Most on-chip networks in practice are general-purpose and use routing algorithms such as dimension-ordered routing and minimal adaptive routing. 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.

These application-aware routings are configured before running the applications and can be implemented as source routing or distributed routing. However, 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.

This becomes a great challenge when these permutation networks need to be implemented under very limited on-chip power and area overhead.

Reviewing on-chip permutation networks (supporting either full or partial permutation) with regard to their implementation shows that most the networks employ a packet-switching mechanism to deal with the conflict of permuted data. Their implementations either use first-input first-output (FIFO) queues for the conflicting data, or time-slot allocation in the overall system with the cost of more routing stages, or a complex routing with a deflection technique that avoids buffering of the conflicting data. The choices of network design factors, i.e., topology, switching technique and the routing algorithm, have different impacts on the on-chip implementation.

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.

Regarding the routing algorithm, the deflection routing is not energy-efficient due to the extra hops needed for deflected data transfer, compared to a minimal routing. Moreover, the deflection makes packet latency less predictable; hence, it is hard to guarantee the latency and the in-order delivery of data. This paper presents a novel silicon-proven design of an on-chip permutation network to support guaranteed throughput of permutated traffics under arbitrary permutation. Unlike conventional packet-switching approaches, our 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.

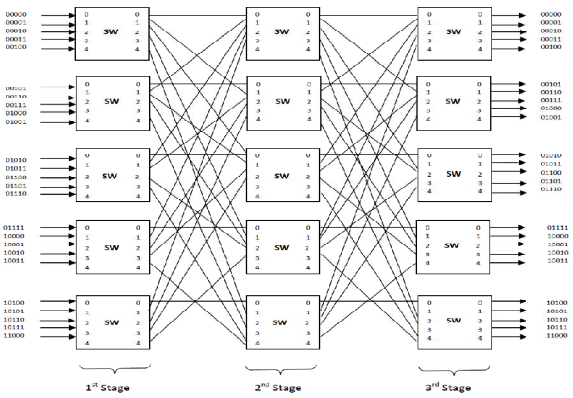


Figure 1: Proposed on-Chip Network Topology with Port Addressing Scheme

II. 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. Before mentioning the dynamic path-setup scheme, the network topology is first discussed. Then the designs of switching nodes are presented.

A. On-Chip Network Topology

Clos network, a family of multistage networks, is applied to build scalable commercial multiprocessors with thousands of nodes in macro systems. 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, we proposed to use C (4, 4, 4) as a topology for the designed network. This network has a rearrangeable property that can realize all possible permutations between its input and outputs. The choice of the three-stage Clos network with a modest number of middle-stage switches is to minimize implementation cost, whereas it still enables a rearrangeable property for the network.

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.

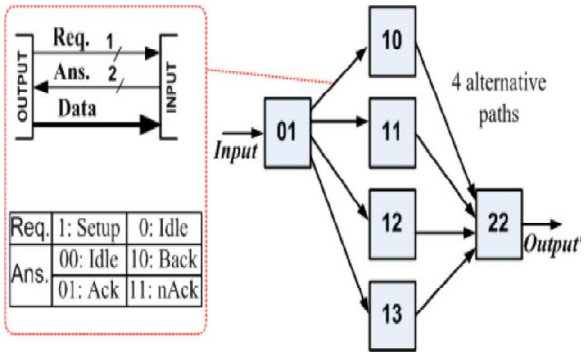


Figure 2: Switch-By-Switch Interconnection and Path-Diversity Capacity

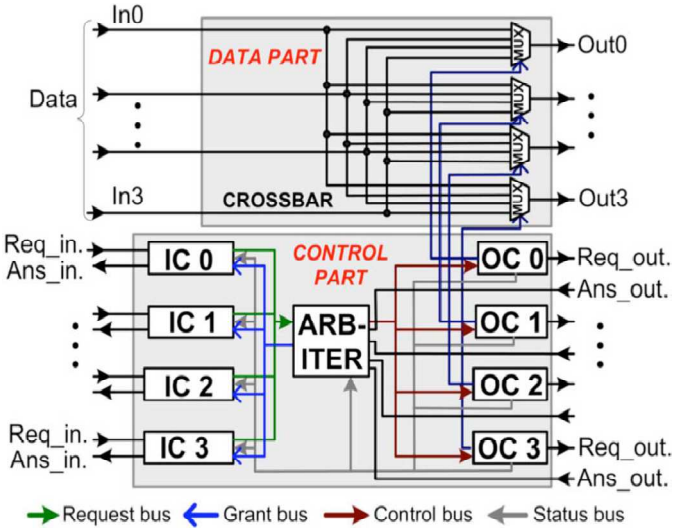


Figure 3: Common Switch Architecture

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 the 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 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 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 = 10(Back) means that the link is blocked. This Back code is used for a backpressure flow control of the dynamic path-setup scheme, which is discussed in the following subsection.

B. 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. Each path setup, which starts from an input to find a path leading to its corresponding output, is based on a dynamic probing mechanism. The concept of probing is introduced in works, in which a probe (or setup flit) is dynamically sent under a routing algorithm in order to establish a path towards the destination. Exhausted profitable backtracking (EPB) is proposed to use to route the probe in the network work. 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. A question is that can the proposed EPB-based path setups used with the Clos C(4, 4, 4) realize all possible full permutations between its inputs and outputs? As proofed in works, the three-stage Clos network C (4, 4, 4) is rearrangeable if $m \geq n$ 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. By the Exhaustive Property of EPB as proofed inwork [12], the EPB-based path setup completely searches all the possible paths within the set of path diversity between an idle input and idle output. Directly applying the Exhaustive Property of the search into rearrangeable C (4, 4, 4) shows that the EPB-based 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 EPB-based 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. As designed in this network, each input sends a probe containing a 4-bit output address to find an available path leading to the target output. During the search, the probe moves forwards when it finds a free link and moves backwards when it faces a blocked link. By means of non-repetitive movement, the probe finds an available path between the input and its corresponding idle output. The EBP-based path-setup scheme is designed with a set of probe routing algorithms as mentioned later. The following example describes how the path setup works to find an available path by using the set of path diversity. It is assumed that a probe 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 02). First, the probe 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 22.

- If link 10 - 22 is available, the probe 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 probe will move back to switch 01 and link 01 - 10 is released. From switch 01, the probe 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 probe can dynamically set up the path in runtime in a conflict-avoidance manner.

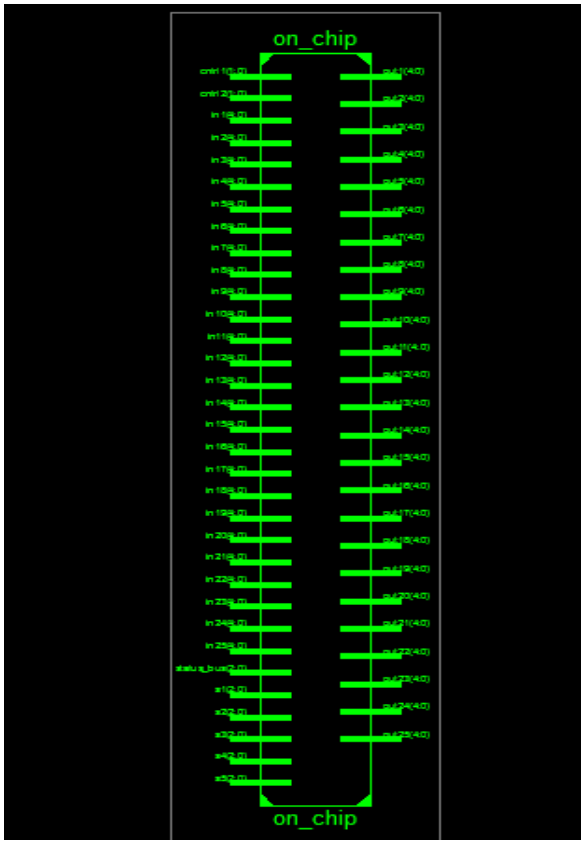
C. Switching Node Designs

Three kinds of switches are designed for the proposed on-chip network. These switches are all based on a common switch architecture, with the only difference being in the probe routing algorithms. This common architecture has basic components: INPUT CONTROLs (ICs), OUTPUT CONTROLs (OCs), an ARBITER, and a CROSSBAR. Incoming probes in the setup phase can be transported through the data paths to save on wiring costs.

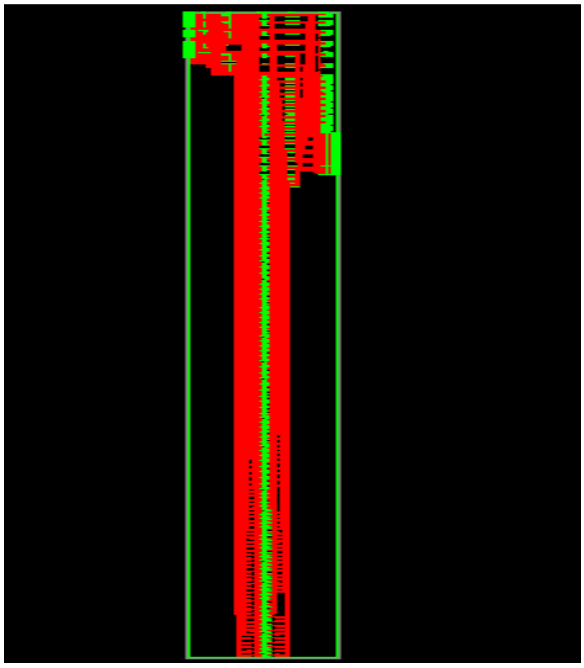
The ARBITER has two functions: first, cross-connecting the Ans_Outs and the ICs through the Grant bus, and second, as a referee for the requests from the ICs. When an incoming probe arrives at an input, the corresponding IC observes the output status through the Status bus, and requests the ARBITER to 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, resolve 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

III. SIMULATION RESULTS

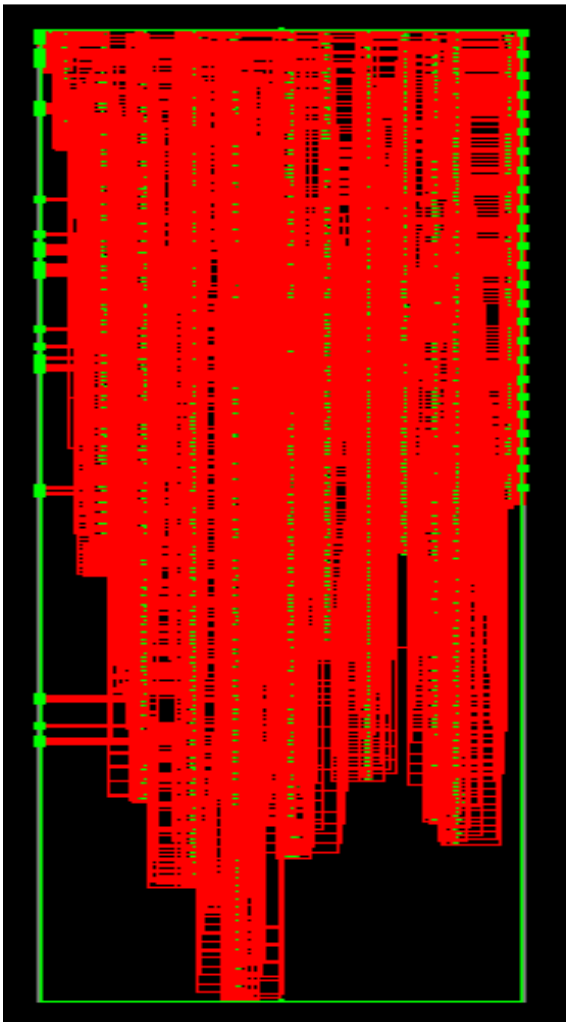
Block Diagram



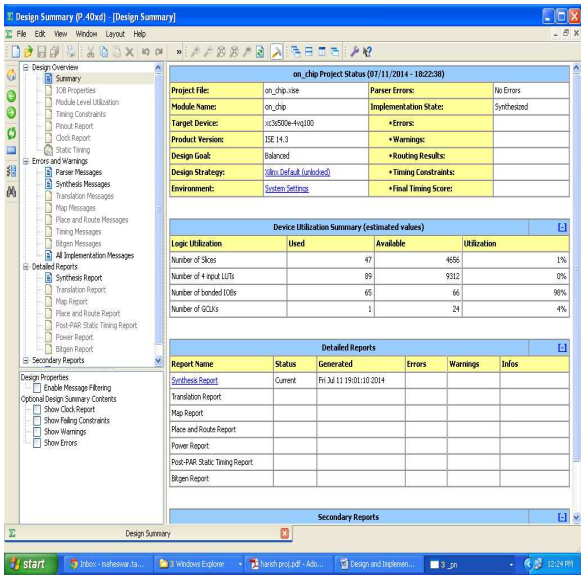
RTL Schematic



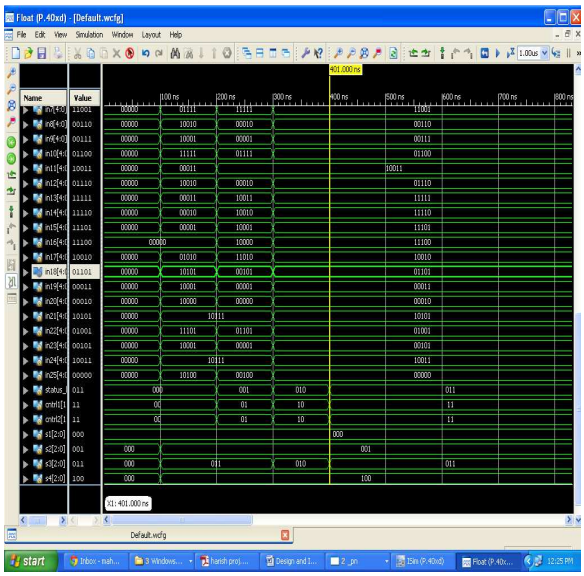
Technology Schematic



Design Summary



Output Waveform



IV. CONCLUSIONS

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 silicon-proven test-chip validates the proposed design and suggests availability for use as an on-chip infrastructure-IP supporting traffic permutation in future MPSoC researches.

REFERENCES

1. S. Borkar, “Thousand core chips—A technology perspective,” in *Proc. ACM/IEEE Design Autom. Conf. (DAC)*, 2007, pp. 746–749.

2. P.-H. Pham, P. Mau, and C. Kim, "A 64-PE folded-torus intra-chip communication fabric for guaranteed throughput in network-on-chip based applications," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, 2009, pp. 645–648.

3. C. Neeb, M. J. Thul, and N.Wehn, "Network-on-chip-centric approach to interleaving in high throughput channel decoders," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2005, pp. 1766–1769.

4. H. Moussa, A. Baghdadi, and M. Jezequel, "Binary de Bruijn on-chip network for a flexible multiprocessor LDPC decoder," in *Proc. ACM/ IEEE Design Autom. Conf. (DAC)*, 2008, pp. 429–434.

5. H. Moussa, O. Muller, A. Baghdadi, and M. Jezequel, "Butterfly and Benes-based on-chip communication networks for multiprocessor turbo decoding," in *Proc. Design, Autom. Test in Euro. (DATE)*, 2007, pp. 654–659.

6. S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, "An 80-tile sub-100-w TeraFLOPS processor in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 29–41, Jan. 2008.

7. W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks:*. San Francisco, CA: Morgan Kaufmann, 2004.

8. N. Michael, M. Nikolov, A. Tang, G. E. Suh, and C. Batten, "Analysis of application-aware on-chip routing under traffic uncertainty," in *Proc. IEEE/ACM Int. Symp. Netw. Chip (NoCS)*, 2011, pp. 9–16.

9. P.-H. Pham, J. Park, P. Mau, and C. Kim, "Design and implementation of backtrackingwave-pipeline switch to support guaranteed throughput in network-on-chip," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 10.1109/TVLSI.2010.2096520.

10. D. Ludovici, F. Gilabert, S. Medardoni, C. Gomez, M. E. Gomez, P. Lopez, G. N. Gaydadjiev, and D. Bertozzi, "Assessing fat-tree topologies for regular network-on-chip design under nanoscale technology constraints," in *Proc. Design, Autom. Test Euro. Conf. Exhib. (DATE)*, 2009, pp. 562–565.

11. Y. Yang and J.Wang, "A fault-tolerant rearrangeable permutation network," *IEEE Trans. Comput.*, vol. 53, no. 4, pp. 414–426, Apr. 2004.

12. P. T. Gaughan and S. Yalamanchili, "A family of fault-tolerant routing protocols for direct multiprocessor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, no. 5, pp. 482–497, May 1995.

13. V. E. Beneˇs, *Mathematical Theory of Connecting Networks and Telephone Traffic*. New York: Academic Press, 1965.