International Journal of Engineering Research and General Science Volume 2, Issue 6, October-November, 2014
ISSN 2091-2730

Design and Implementation of an On-Chip timing based Permutation Network for Multiprocessor system on Chip

Ms Lavanya Thunuguntla[1], Saritha Sapa [2]

1 Associate Professor, Department of ECE, HITAM, Telangana (S), India

2 Research Scholar, Department of ECE, HITAM, Telangana (S), India

Abstract: The communication mechanism is employed in systems on a single chip (SoC) are an important contribution to their overall performance. To date bus based mechanism is applied in many areas of real time applications of SoCs realizing on FPGA due to its flexibility and simplification in designing tool. This paper presents a novel on-chip network to support traffic permutation in multiprocessor SoC applications. The proposed work employed dynamic path setup with round robin algorithm with word length control and also circuit switching approach to enable runtime path arrangement for arbitrary traffic permutations. The proposed work is carried out based on Xilinx FPGA family and by using Verilog HDL

Keywords: SoC, FPGA and Dynamic path setup

I. INTRODUCTION

Today field programmable gate-arrays (FPGAs) are used for a wide sector of applications. The usage in former times was focused on rapid-prototyping system for integrating test systems. After the test-phase, often an ASIC approach substituted these systems for mass-production. Due to dramatic growth in circuit-design complexity regarding Moor's Law, the ability of implementing complex architecture in a single chip always presents new challenges. One of the issues found by designers while implementing large SoCs is the communication among their components. Buses are an increasingly inefficient way to communicate, since only one source can drive the bus at a time, thus limiting bandwidth.

SoCs are increasing in popularity because of their advantages: larger bandwidth, and lower power dissipation through shorter wire segments. Communications in large SoCs are so important that many designers have adopted the NoC approach. The challenges consist in offering the best connectivity and throughput with the simplest and cheapest architecture of methodology; whereas, many topologies and architectures have been investigated. This is well illustrated in [4], where researchers propose a two-level FIFO approach in order to simplify the design of the arbitration algorithm and improve the bandwidth. However, this method tends to be expensive in term of hardware.

Although the completely embedded tools of FPGA manufactures such as Xilinx and Altera are offered to help their customers to design the complex Multi Processor Systemon- Chip (MPSoCss), their environments only offer the busbased paradigm or point-to-point connection. More complex MPSoCs may require higher bandwidths than a bus-based system can offer, or may need to be more efficient than pointto- point connections.

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. The idea of designing the Reconfigurable Crossbar Switch for NoCs to gain a high data throughput and to be capable of adapting topologies on demand was presented in [5]. Their evaluation results showed that output latency, resource usage, and power consumption were better than a traditional crossbar switch. Nevertheless, they did not focus the problem while operating many-to-one and one-to-many data communication.

This paper is organized as follows: Section II presents over communication problems; section III presents on-chip network with dynamic path setup; section IV implementation of on-chip network and V section is for concluding the work.

II. COMMUNICATION PROBLEM

In Fig. 1, all communication in a process group becomes many communications involving an arbitrary subset of processor from the system's point of view. In order to efficiently support data communication, a system has to support one-to-one (unicast), one-to-many (multicast), many to- one (gathering) and many-to-many communication primitives in hardware. Actually, most interconnection can support unicast, but not gathering and multicasting.

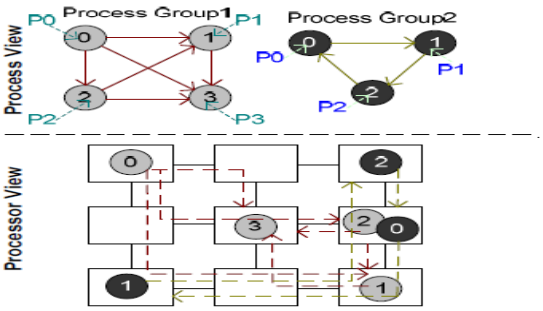

Fig. 1. System's point of view

In Fig. 2(a) shows a multicast communication with three destinations where process P0 has to send the same data to three processors: P1, P2 and P3. Without the multicast functionality on interconnection, the system can use unicast functionality to send data to all destinations sequentially. However, blocking will occur, if P0 is executing sending state to P1, and P1 has not yet execute receiving state, P0 is blocked; meanwhile, P2 is executing receiving state, and is blocked because P0 has not yet executed sending state. Obviously, system resources are wasted due to unnecessary blocking. Fig.2(b) shows gathering data communication from three sources to one destination. Because of sending data from all sources, congestion is found at destination where it can be reduced by applying source priority. Supposing P1,P2 and P3 are the first, second and third priorities. While their data arrive at P0, the data from P1 will be the first forwarding; meanwhile, the rest will be buffered. Thus, the output latency of system will increase, and buffer will also require.

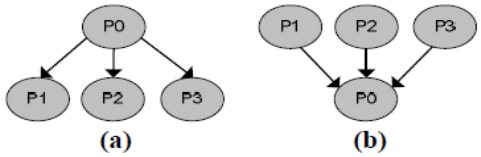

Fig. 2. a) One-to-many b) Many-to-one

III. ON CHIP NETWORK DESIGN

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.

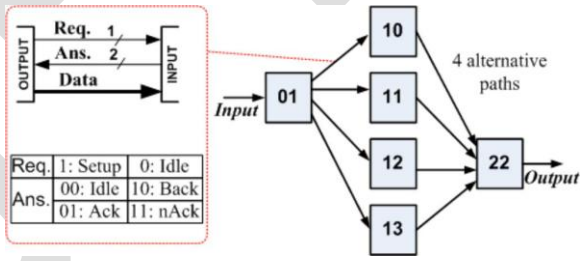

Fig. 3. Switch-by-switch interconnection and path-diversity capacity.

Fig. 3. The bit format of the handshake includes a 1-bit *Request (Req)* and a 2-bit *Answer (Ans)*. Req=1is 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 (ack) 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 (nack) 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 (*Block*) means that the link is blocked. This *Block* code is used for a backpressure flow control of the dynamic path-setup scheme.

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 shown in Fig. 4, 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, resolves 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. The IC is implemented with finite-state machine (FSM). The probe routing algorithm and the operation of the switches are controlled according to this FSM implementation.

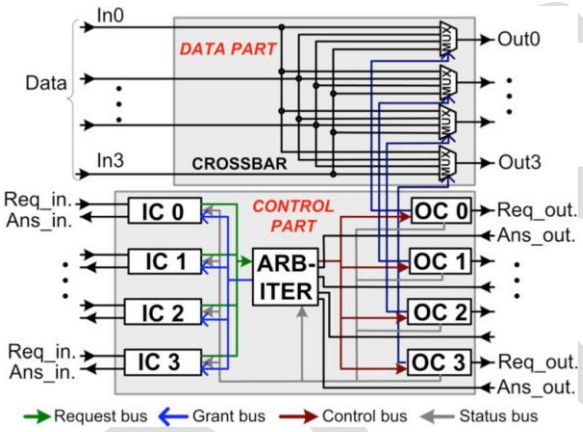

Fig. 4. Common switch architecture.

IV. IMPLEMENTATION OF ON-CHIP NETWORK

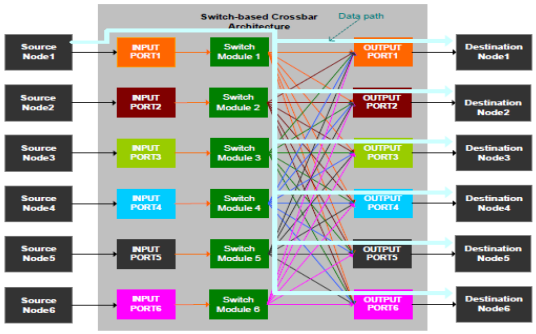

Fig. 5. 6x6 switch-based crossbar structure.

The major components that make up the switch-based crossbar consist of Input Port module, Switch module and Output Port module. Depending on the kind of channels and switches, there are two alternatives for designing switch-based crossbar: unidirectional and bidirectional [6]. In this paper, unidirectional switch-based crossbar is selected and simplified with 6x6 switch-based crossbar structure shown in Fig. 1 because of resource constraint. Obviously, transmitting data from a Source Node to a Destination Node require crossing the link between the Source Node and the Input Port module, and the link between the Output Port module and the Destination Node where the Switch module in data path will dynamically establish the link for the Output Port module according to switching protocol.

According to the 6x6 switch-based crossbar architecture, its switching protocol shows in Fig. 5.
Because of resource constraint on the target FPGA [2], data width, the N.of word register, the destination port register are 16,10 and 6 bits respectively. Moreover, synchronous protocol is applied to synchronize all modules in the switch-based crossbar-Clock signal, Ready signal and acknowledge signal.

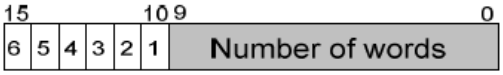

Fig. 6. 16-bit switching protocol

Fig.6 shows the 10-bit Last Significant Bit (LSB) of switching protocol, which defines the number of packets, required transferring from a Source Node to a Destination Node, where the maximum is 1,024 packets per time, and the rest define the Destination Node. For example, when the Source Node1 wants to transfer 100 packets to the Destination Node number 3 and 4, the 16-bit switching protocol has to be 0011_0000_0110_0100 (0x3064H).

Input Port module

Its behavior shows in Fig. 7 based on switching conceptual [1]. At the beginning, the state is in an idle state, and then the header ready signal enables HIGH to inform a Source Node that ready to read the switching protocol. As soon as the 16-bit switching protocol is asserted at the Data in signal and the Data in valid signal, the state goes to the next check state. In this state, the 16-bit switching protocol is separated and written on the N.of word register and the destination port register resided in FSM module, where 10-bit low and 6-bit high are written on the N.of word register and the destination port register; meanwhile, the register port signal is read to check, whether or not the required destination ports are free.

Fig. 7. State machine

If it is free, the state will go to the req state. In the req state, the N.of words register, the destination port register and the req signal are read and presented on the N.of words signal and the N.of word valid signal. When the rsp signal enables HIGH, the channel for transferring data is guaranteed, and the state goes to the Run state. In the Run state, the data in ready signal enables HIGH. The 16-bit data in signal and the data in valid signal are sequentially asserted into
the channel. When the lasts data is completely forwarded, the rsp signal from the Output port module will enable LOW, and the state will start to the next cycle.

Output Port module

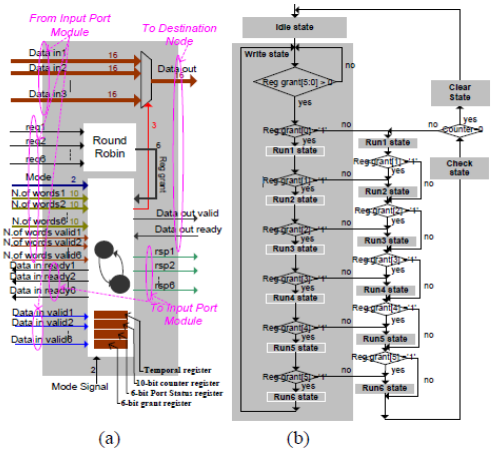

Fig. 8. a) *Output Port module* Architecture
b) Flow control diagram

Fig. 8(b) explains its behavior. At first, the state is in an Idle state, and the mode signals are read where there are two kind of possible modes, normal mode and interleaving mode. In the normal mode, the Round Robin component will be enabled but will be disabled in the interleaving mode. Whenever a Source Node requires transferring its data to a Destination Node, the req signal of the Input Port module connected with the Source Node will enable HIGH. Then, at the Output-Port module connected with the Destination Node, each bit of the 6-bit grant register related with each req signal of each Input Port module will set "1". The N.of words signal and the N.of words valid signal will be read and stored into the temporal register. Then, the state goes to the Write state. In the Write state, each bit of the grant register is read to check the Source Node's requirement and to identify the location of the next state. In order to simplify, supposing the grant register contains 101100;

It implies that the input port modules number 3,4 and 6 will forward their data to the output port module. Therefore, the state will start at the Run3 state, and the Data in ready signals number 3,4 and 6 will be enabled HIGH. Until the Run6 state is executed, the state goes to the Check state. The counter register counting the forwarded packets is checked, whether or not the forwarded packets has been completely sent. Whenever, they have been sent properly this counter register will be "0000000000" and the Clear state will be done. At the Clear state, the rep signals at 3,4 and 6 are enabled LOW and the grant register will be "000000", as well as the state will begin the next cycle.

V.CONCLUSION

This paper proposes and implements the interleaving mechanism to overcome the output delay (latency) while operating many-to-one and one-to-many data communication. All primitive modules are structured by Verilog HDL, verified their behaviors with Model Sim 6.2c, synthesized their resource usages and estimated frequency on the Xilinx FPGA by ISE tool.

REFERENCES;

[1] D. Bafumba-Lokilo, Z. Savaria, J. David, "Generic Crossbar Network on Chip for FPGA MPSoCs", Circuits and Systems and TAISA Conference, 2008, pp 269-272.
[2] Xilinx , "On-chip Peripheral Bus V2.0 with OPB arbiter", "Processor Local Bus(PLB)v4.6(v1.03a)", http://www.xilinx.com.
[3] H. C. d. Freitas, P. Navaux, "On the Design of Reconfigurable Crossbar Switch for Adaptable On-Chip Topologies in Programmable NoC Routers", ACM Special Interest Group on Design Automation, 2009, pp. 129-132.
[4] H.Po-Tsang, H. Wei , "2-Level FIFO Architecture Design for Switch Fabrics in Network-on-Chip", Circuits and Systems, ISCAS 2006, Proceedings 2006, IEEE International Symposium on, pp.4863-4866.
[5] M. Hübner, L. Braum,D. Göhringer, J. Becker, "Run-Time Reconfigurable Adaptive Multilayer Network-On-Chip for FPGA-Based systems", IEEE International Symposium on In Parallel and Distributed Processing, 2008, pp. 1-6.
[6] C. Hilton, B. Nelson, "PNoC: a flexible circuit-switched NoC for FPGA based systems", IEE Proceeding Computing Vol. 153, 2006, pp. 181-188