ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

An Efficient Approach of Designing High Speed Dynamic Routing of Switching Networks on a Single Chip for Multiprocessors Communication

M Nagesh
Swarnandhra College of Engineering and Technology, Seetharampuram, Narsapur.

J E N Abhilash
Swarnandhra College of Engineering and Technology, Seetharampuram, Narsapur.

N Srikanth
Swarnandhra College of Engineering and Technology, Seetharampuram, Narsapur.

Abstract:

This paper gives an FPGA architecture implementation of novel-on-chip network for guaranteed traffic permutation used when multiprocessors are connected on a single chip. This proposed architecture works with a pipelined circuit switching approach combined with dynamic path setup scheme under different stage network topology. This new dynamic path setup approach scheme helps in runtime path arrangement for any arbitrary traffic permutations. The main advantage of this architecture it combines the path architecture of circuit switching which gives a combined transmission of data and also dynamic path setup helps to change its path automatically so final destination will be reached from one transmitted to only one receiver in around any one of four paths. In this we proposed Clos architecture of 4x4x4 size so three stages complemented.

Index Terms:

Throughput, multistage interconnection network-on-chip, pipelined circuit switching, traffic permutation.

I. INTRODUCTION:

Due to the enhancement in the VLSI technology multiprocessor fabrication on a single chip has been developed. These multiprocessors require good communication path between them. Mainly the major drawback of multiprocessor system is traffic overloading which results a blockage of paths. Now a day's multiprocessor system on chips (MPSoc) interconnected with on chip networks plays a vital role in applications of parallel processing, scientific computing, and high speed networks and so on.

Traffic in the routing paths must be organized in a special manner to achieve high speed communication. 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 imperative traffic classes exhibited from on-chip multiprocessing applications.

so that unnecessary blockage of the packet data. Standard permutations of traffic happen in general MPSoCs, for example, polynomial, sorting, and FFT computations cause shuffled permutation, recently, application-specific MPSoCs targeting flexible Turbo/LDPC decoding have been developed, and they produce arbitrary and concurrent traffic permutation and combination due to many mode and standard feature.

In addition to many of the MPSoC applications 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 networks using application-aware routings are needed to obtain improved performance compared to the general 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 handle the dynamic changes of permutation traffic patterns, which is exhibited in many of the application phases. The complexity lies in the design endeavor to compute the routing to support the permutation changes in execution time, as well as to assurance 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.

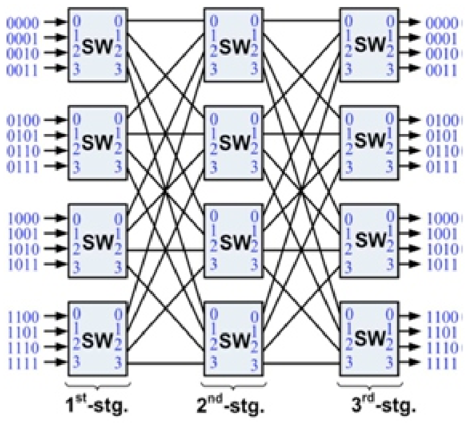

Fig.1. Proposed on-chip network topology with port addressing scheme

Regarding the on and off technique, packet switching requires an more amount of on-chip power and space for the queuing buffers (FIFOs) with pre-calculated queuing depth at the switching nodes and/or network interfaces. Regarding the routing algorithm, the deflection routing is not sufficient energy 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.

Unlike conventional packet-switching approaches, our on-chip network employs a circuit-switching mechanism with a dynamic route setup scheme under a multi stage 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 output assurance. By removing the extra load overhead of queuing buffers, a compact implementation is obtained and stacking multiple networks to support concurrent permutations in runtime is feasible.In the rest of the paper in section II we mainly focus on proposed on chip network design to support runtime path arrangement. Section III provides all the execution particulars as proof of notion test switch. Section IV enhances the implementation, and finally section V ends this paper and provides all the outlines as a future scope to further researches.

II. PROPOSED ON CHIP NETWORK DESIGN:

As from the concept in section I the main idea behind the proposed on chip network design is based on pipelined circuit witching approach with a dynamic path-setup scheme supporting runtime path arrangement.

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

1.On-Chip Network Topology:

Networks-on-chip (NoCs) closely resemble the interconnect architecture of high-performance parallel computing systems. For this reason, the interconnection topologies used in the early NoC prototypes can be traced back to the field of parallel computing. The proposed on chip network is Clos network. Clos network, a family of multistage networks, is applied to build many macro systems with thousand nodes having multiprocessors. A typical three-stage Clos network is defined as C (r, s, t), where r represents the number of inputs in each of t first-stage switches and s is the number of second-stage switches.

In order to support parallel working degree of 16 as in most practical MPSoCs, we proposed to use C (4, 4, 4) as a topology for the designed network (see Fig. 1). 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 are to minimize cost, whereas it still enables a reconfigurable property to the network.

There are basically three types of switching methods are made available. Among the three, circuit switching and packet switching are commonly used but the message switching has been opposed out in the general communication procedure but is still used in the networking application. 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. To support this circuit-switching scheme, a one to one interconnection with its handshake signals is proposed.

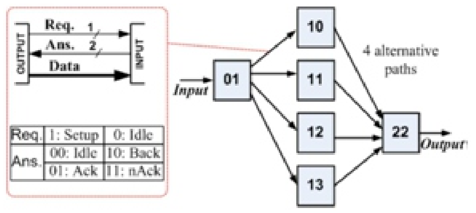

Fig.2. Switch-by-switch interconnection and path-diversity capacity.

As shown in Fig.2 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 (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 created, then a data transfer can be started instantaneously. 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 (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.

2.How the path is arranged:

The scheme used is dynamic path-setup scheme and 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 snooping 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. A path pact with full permutation consists of sixteen path setups, whereas a path deal with partial permutation may consist of a detachment 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 source and destination? As shown in works, the three-stage Clos network C (r,s, t) is configurable if s ≥ r. In the proposed network of C(4, 4, 4). s=r=4, so it is reconfigurable.There always exists an available passageway from an unoccupied input leading to an unoccupied output. By the Exhaustive Property of EPB as considered in work, the EPB-based path setup completely searches all the possible paths within the set of path diversity between an unoccupied input and unoccupied output.

Directly applying the Exhaustive Property of the explore into rearrangeable C(4, 4, 4) shows that the EPB-based path association 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 plank output. During the search, the probe moves in forward direction when it finds a free link and moves backwards when it faces a barren link. By means movement with no repetition, the query finds an available path between the input and its corresponding unoccupied output. The EBP-based path-setup method 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 shown in Fig. 2. It is assumed that a query from a source (e.g., an input of exchange 01) is trying to set up a path to a board destination (e.g., an on hand output of exchange 22). First, the probe will non-repetitively try paths through the second-stage switches in the order of10›11›12›13. Assuming that the link 01–10 is available, the probe first tries this link (Req=1) and then arrives at switch 10.

- If link10–22 is available, the probe arrives at switch 22 and meets the board 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(Ans=Back) and link 01–10 is released (Req=0). 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 barren links and trying others, the probe can dynamically set up the path in runtime in a conflict-avoidance manner

3.Architecture Of Switching Nodes:

Out of all the available three switching nodes the switching nodes used in our proposed on chip network are all based on common switch architecture as in fig.3, with the only difference being in the query routing algorithms. This general architecture has basic apparatus:

INPUT CONTROLs (ICs), OUTPUT CONTROLs (OCs), an ARBITER, and a CROSSBAR. Incoming probes in the set-up phase can be transported through the data paths to save on wiring costs.

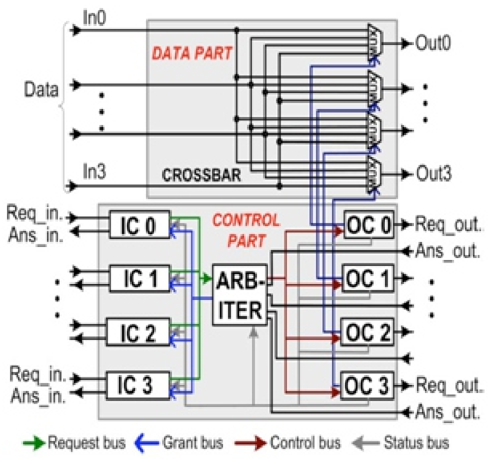

Fig.3. Common switch architecture.

The ARBITER has two functions: first one is, cross-connecting the Ans_Outs and the ICs through the Grant bus, and second, as a arbitrator for the requests from the ICs. When an incoming query reaches at an input, the corresponding IC observes the output status all the way through the Status bus, and requirements the ARBITER to grant it admission to the corresponding OC through the Request bus.

When accommodating 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-considered priority rule, resolves contention when several ICs request the same free output. After this resolution, only one IC is accepted, whereas the rest are replied as facing a blocked link (i.e., similar to receiving an Ans=Back).

The OCs work as re-timing stages for the commands from ARBITER placed on the Control bus and control the CROSSBAR. The CROSSBAR is a 4 x 4 full-connecting matrix designed with output multiplexers. The ICs and the ARBITER are clocked with the rising and the falling edges of the clock, respectively. By this realization, snooping is dynamically processed by the switch in one clock cycle basis. As denoted in Fig. 3, the control part of switches performs the dynamic EPB-based path setup, whereas the data part just provides configured paths for sure circuit-switched data. This meets the intention of designing the circuit-switched switches to support EPB-based path setup in C(4,4,4) network.To validate if the designed network works as desired, a test bench is applied to test the capability of realizing full permutation with sixteen path setups. To avoid a path setup interfering with others during the search and incurring a rearrangement of existing paths, a delay is set between the path setups launched one-by-one in a sequence in the test bench. This is to ensure that the previous path setup is completed before a new one is launched. As considered based on the path-diversity graph shown in Fig. 2, the worst-case path setup needs 14 steps (hops) of moving its probe back and forth to search for a path. Each step of moving the query needs two cycles, as derived from the cycle-accurate design model.

4.Working with an Example:

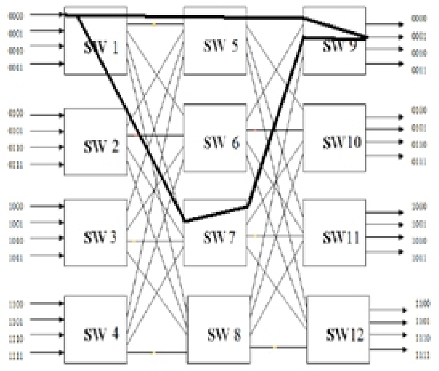

Fig.4 Example to create and select the path to transfer data

As shown in the Fig.4 whenever the data is need to be transferred through an on-chip network firstly a path has to be created in the next instance the path need to be authorized for data transfer. In our example we tried to transfer the data from input port 0000 to output port 0001. Initially a path has to arrange so that there is no traffic at all as shown I have tried to create a path through 0000 port of first switch to output but from fig.4 that path is not free to move the data in the forward direction. So it is mandatory to come back to port 0000 again I have to select another path to transfer the data. Now our proposed design dynamic path setup arrangement came into picture to select and authorize another path to transfer the data. In our example the path dynamically chosen is through switches 1.7,9. In this way our proposed design can transfer the data in the fastest manner from input to the output.

III. IMPLEMENTATION DETAILS:

The design is implemented on Xilinx ISE 9.1 the data is transferred from one stage to another stage by having an acknowledgement from the next stage. In order to transfer the control from one stage to another stage In the below simulation wave forms we have shown how the data is transferred from input to output based on the status of control signals (req_in, ans_in, input, req_out, ans_out) on the switch. The proposed network design is having a data of width 4 bits. There are totally 256 paths in the entire network. One input will reach to the final output in 16 different paths.

Each switch is verified individually using VHDL programming and with help of test benches. Initially we have to send a request depending on the available paths that means ans_outs of the code outputs it will select one output line. We have taken only priority based approach where request0 is a highest priority and followed by request1, request2, request3. When out of four paths first path is failed then only it verified for second path and so on this process is continued until the source has to reach the required destination here we have seen that our proposed design is automatically changing its path according to the acknowledge and negative act given by the respective switches. We have observed that our approach is having ability to sustain heavy traffic and automatically changes its path.

RESULTS:

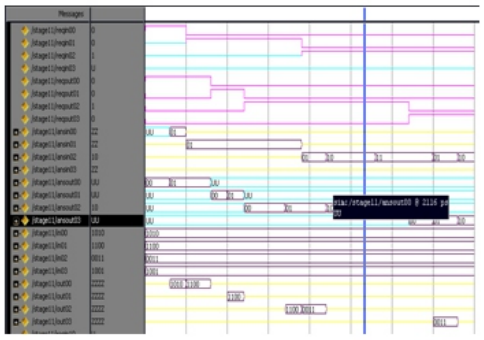
Fig.5. Stage One Output Wave Form

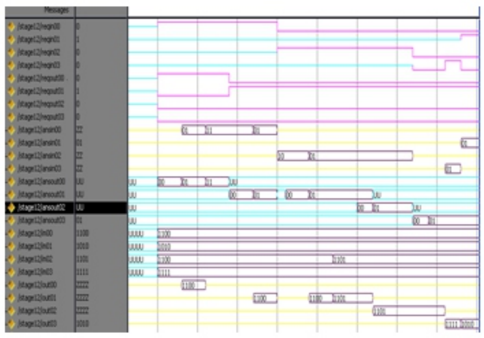
Fig.6. Stage Two Output Wave Form

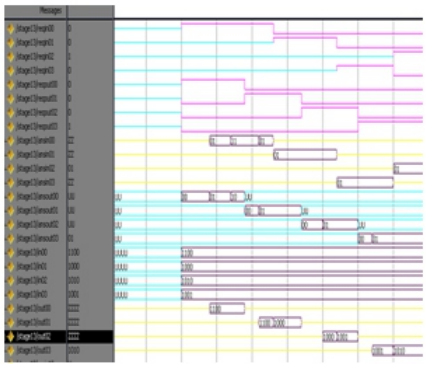
Fig.7. Stage Three Output Wave Form

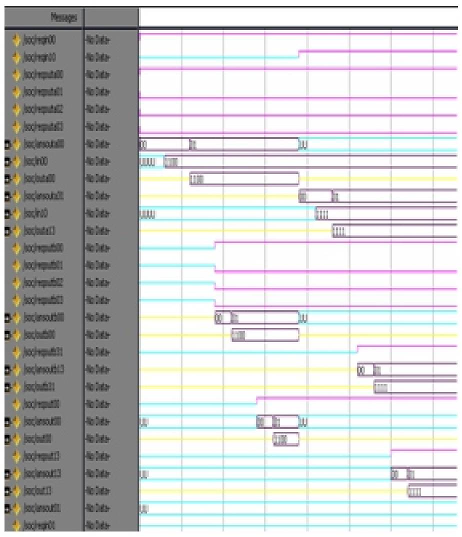
Fig.8. Final Output Wave Form

IV.CONCLUSION:

This paper has analyzed for the best way in supporting on chip network which over comes the problem of traffic permutation in soc applications. This approach is a combination of traffic path setup schemes and new network topologies is called Clos.

Our proposed design was implemented using VHDL programming and with the help of test benches. We had verified all the combinations i.e. for 4x4x4 Clos network all 256 have been verified. This can be supportable for fabrication an on Soc design for the multiprocessor communications (Mpsoc).

REFERENCES:

[1] Phi-Hung Pham, Junyoung Song, Jongsun Park, and Chulwoo Kim"Design and implementation of an on chip permutation network for multiprocessor system on chip" IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 21,pp. 173-177.

[2] S. Borkar, "Thousand core chips—A technology perspective," in Proc. ACM/IEEE Design Autom. Conf. (DAC), 2007, pp. 746–749.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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 TeraFLOPSProcessor in 65-nm CMOS," IEEE J. Solid-State Circuits, vol. 43, no. 1, pp. 29–41, Jan. 2008.

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

[9] 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.

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

[11] 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.

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

[13] 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.

[14] V. E. Beneˇs, Mathematical Theory of Connecting Networks and TelephoneTraffic. New York: Academic Press, 1965.