3K-05

A Novel Time-Division Multiplexing Approach for Emulating NoC Architectures on FPGAs

Thiem Van Chu † Kenji Kise †

† Graduate School of Information Science and Engineering, Tokyo Institute of Technology

1 Introduction

Network-on-Chip (NoC) is becoming mainstream in many-core designs. It is believed that the near future many-core designs will have thousands of cores interconnected by NoC architectures. To investigate performance trade-offs in various aspects of NoC designs, software-based simulators such as Booksim [1] have been widely used. However, their simulation speed decreases rapidly with increasing the number of NoC nodes.

Since FPGAs provide a highly parallel platform, FPGA-based simulation is a promising approach to the simulation speed problem. It is, however, difficult to fit an NoC architecture with thousands of nodes to a single FPGA. Although multiple FPGAs can be used, this approach not only leads to a higher cost, but also makes the design become much more complex. Moreover, the off-chip communication becomes the performance bottleneck of the entire system.

This paper proposes a novel time-division multiplexing (TDM) approach which enables architects to fast and accurately emulate large-scale NoC architectures using a single FPGA. For a given NoC architecture, instead of directly implementing it on an FPGA, we effectively utilize FPGA resources to emulate the behavior of the entire network using one or several physical nodes.

2 FPGA-based NoC Emulation and TDM

Although FPGA-based simulation is a compelling approach, scaling an FPGA-based NoC emulator to thousands of nodes is a challenging task due to the limitation of FPGA resources. To reduce the FPGA resource usage, the TDM technique can be used. By sharing the combinational logic between virtual instances, large amount of Look-up Tables (LUTs) can be saved. Thus, the number of required slices can be reduced significantly.

In FPGAs, there are two types of on-chip memory: block RAM and distributed RAM. A block RAM typically has a capacity of tens of Kilo-bits and cannot be subdivided. Thus, a whole block RAM is required even if a small fraction of it is used. On the other hand, distributed RAMs are more flexible. However, since distributed RAMs are based on LUTs, they are inefficient in term of area for implementing large memory.

The TDM technique can help to utilize block RAMs much more efficiently. Without using the TDM technique, a block RAM cannot be shared between different routers. Since the size of a block RAM is very large, it is very inefficient. On the other hand, multiple routers can share a same block RAM when the TDM technique is adopted. This is important because the number of block RAMs on an FPGA is generally small. For instance, a Spartan-6 LX45 FPGA has only 116 18Kb block RAMs.

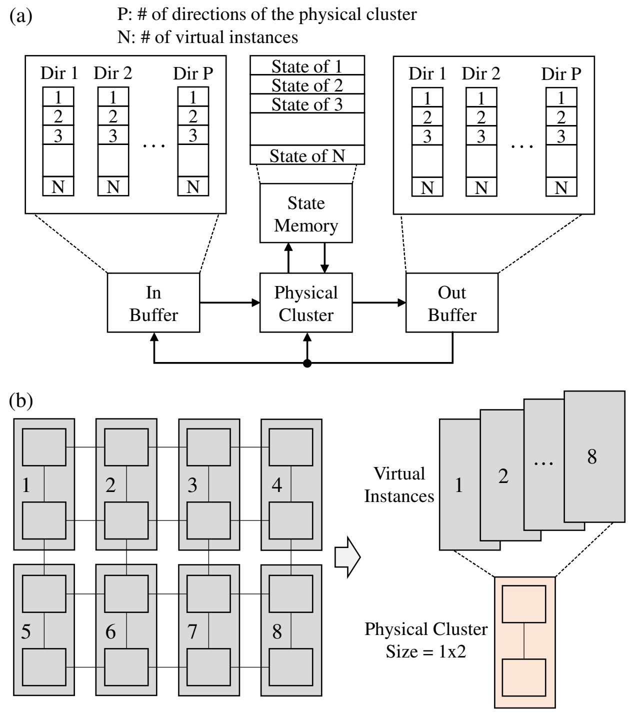

Figure 1: (a) Datapath of the proposed TDM approach and (b) Example of emulating an NoC design with physical cluster size = 1×2.

Fig. 1(a) shows the datapath of the proposed TDM approach consisting of four components: physical cluster, state memory, out buffer, and in buffer. Basically, the physical cluster is composed of one physical node. However, in many cases, we can use multiple interconnected physical nodes to emulate the behavior of the network. For instance, to emulate a 2D mesh network as shown in Fig. 1(b), two physical nodes can be used. There is a trade-off in increasing the number of physical nodes. More physical nodes will improve the simulation speed but consume more hardware resources. To emulate one cycle of the network (emulation cycle), the physical cluster sequentially emulates a determined number of virtual instances. In the example in Fig. 1(b), there are a total of eight virtual instances.

The state memory is used to store states of all virtual instances. The physical cluster emulates different virtual instances by using different state data loaded from the state memory. Also, when the emulation of a virtual instance is finished, the state of the virtual instance in the state memory is overwritten by the new state. The state updated at emulation cycle i will be used at emulation cycle $i+1$. The state memory can be implemented using block RAMs because we read only one entry and write only one entry of it at each FPGA cycle.

To resolve the data dependency between virtual instances, we adopt a double-buffering scheme with the out buffer and the in buffer. To emulate a virtual instance, in addition to the state data read from the state memory, the physical cluster needs appropriate data from other virtual instances. For example, in Fig. 1(b), emulating virtual instance 2 in emulation cycle i requires a part of data produced by virtual instance 1 and 3 in emulation cycle $i-1$. The out buffer stores data produced by all virtual instances. Before overwriting data of a virtual instance to the out buffer, we copy the old data of the virtual instance to the in buffer. This data will be used by the subsequently emulated virtual instances. Because a virtual instance may depend on data from several other virtual instances (depend on at most one virtual instance per direction), we divide the out buffer along the directions of the physical cluster as shown in Fig. 1(a). By this way, memory with multiple read ports, which is very expensive to implement on FPGA, can be avoided. As a result, the out buffer can be implemented using block RAMs.

Basically, we have to copy all data from the out buffer to the in buffer. However, some topologies such as 2D mesh allow us to optimize the amount of data needed to be copied by choosing an appropriate emulation order. In particular, the emulation order in the example in Fig. 1(b) requires us to copy only data at the east direction and the south direction (instead of all four directions) from the out buffer to the in buffer. Since there is no virtual instance which has less than two dependent ones, this emulation order is the optimized one.

3 Evaluation

We use the proposed TDM approach to implement an NoC emulator on an Atlys Spartan-6 FPGA Development Board. The emulation model and the method for accurately modeling network traffic are presented in our previous work [2]. Table 1 shows the emulated NoC architecture and configuration parameters. The emulated NoC architecture has 1,024 nodes with the state-of-the-art 5-stage pipelined VC router architecture.

Table 2 shows the implementation results in three cases: physical cluster size = 1×1, 1×2, and 2×2. Xilinx ISE design suit 14.7 is used. We set the Optimization Goal as Speed and the Optimization Effort as High. Also, the timing constraint is set to 50MHz. The implementation results shows that block RAMs are efficiently utilized to implement the state memory, the out buffer, and the in buffer, which are presented in Section

Topology	32×32 mesh
Router architecture	5-stage VC router
Routing algorithm	Dimension-order (XY)
Flow control	Credit-based
VC/Switch allocator	Separable output first
Arbiter type	Fixed priority
Flit size	22-bit
Number of VCs per port	1
VC size	4-flit
Packet length	8-flit
Injection process	Bernoulli process
Traffic pattern	Uniform random
Source queue size	8-flit

Table 1: The emulated NoC architecture and configuration parameters.

	1-phy		2-phy		4-phy	
	#	%	#	%	#	%
Slices	974	14%	1,847	27%	3,155	46%
Slice Reg	1,094	2%	1,944	3%	3,741	6%
LUTs	2,587	9%	4,947	18%	9,339	34%
BRAM (16Kb)	79	68%	74	63%	28	24%
BRAM (8Kb)	13	5%	16	7%	116	50%
Freq (MHz)	50		50		50	

Table 2: Implementation results: physical cluster size = 1×1 (1-phy), 1×2 (2-phy), and 2×2 (4-phy).

2. Also, the proposed TDM approach helps to utilize block RAM much more efficiently. This can be seen in the following simple estimation. Without using the TDM approach, the number of block RAMs required to implement only input buffers of 1,024 routers in the emulated NoC design would be $1 \times 5 \times 1,024 = 5,120$. This number is much greater than the number of block RAMs required to implement the whole sytem using the proposed TDM approach.

4 Conclusion

We discussed an effective approach of using TDM to emulate large-scale NoC architectures on FPGAs. One or several physical nodes can be used to emulate the behavior of the entire network by sequentially emulating virtual instances. We showed that the double-buffering scheme, which is adopted to resolve the data dependency between virtual instances, can be optimized based on the network topology and emulation order. Using the proposed TDM approach, we demonstrated the implementation of an NoC design containing 1,024 nodes with the conventional VC router architecture on a single Spartan-6 LX45 FPGA.

References

[1] Nan Jiang Becker, D.U. ; Michelogiannakis, G. ; Balfour, J. ; Towles, B. ; Shaw, D.E. ; Kim, J. ; Dally, W.J. A detailed and flexible cycle-accurate Network-on-Chip simulator IEEE ISPASS, Apr. 2013.

[2] Chu, Thiem V. and Sato, S. and Kise, K. Challenge for Ultrafast 10K-Node NoC emulation on FPGA IEICE Technical Reports RECONF2014-21, Sep. 2014.