Performance Evaluation of TOFU System Area Network Design for High-Performance Computer Systems

P. BOROVSKA, O. NAKOV, S. MARKOV, D. IVANOVA, F. FILIPOV
Computer System Department
Technical University of Sofia
8 Kliment Ohridski Boul., 1756 Sofia
BULGARIA
pborovska@tu-sofia.bg, nakov@tu-sofia.bg, markov@acad.bg, d_ivanova@tu-sofia.bg,
pilif.pilif@googlemail.com, http://cs-tusofia.eu

Abstract: - Switch and system area network architectural designs are significantly influenced by next generation high-performance computer systems and supercomputer technology. As technology evolves, this impact on interconnection network needs to be considered and reevaluated. In this paper we evaluate a communication performance of a high-speed switch and TOFU system area network design by means of network simulations using OMNET++ simulator. The models under investigation have been verified on the basis of parallel program implementations (C++, MPI) on IBM Blade Center.

Key-Words: - Switch Architectural Design; TOFU System Area Networks; Simulation Model; OMNET++, Communication Performance

1 Introduction

The scientific and technological challenges in modern world demand powerful computational resources like a large numbers of processing cores that are highly interconnected via a high-speed system area network (SAN). SAN offers an attractive solution to the communication crisis and are becoming pervasive in high-performance computer systems (HPCS). A well-designed SAN makes efficient use of scarce communication resources — providing high throughput, low-latency communication between clients with a minimum of cost and energy, [1, 2, 5].

In this paper, we propose a high-speed switch design and perform evaluation of system area networks (SAN) via simulations using the discrete event simulator OMNeT++, [4].

Simulation experiments are intended to model a high-speed switch and SAN design for high-performance computer systems, which connect nodes in TOFU topology and evaluate the communication efficiency impact on different traffic patterns.

The communication performance parameters are estimated on the basis of parallel simulation models in OMNeT++ network simulator environment which have been run on IBM Blade Center.

2 Switch design

We suggest a switch architecture that has a highly regular structure. The switch architecture for the case of four input and output ports is shown in Fig. 1.

The proposed switch design comprises identical modules for each communication transfer. It has an input multiplexer, which selects either direction from neighbor switch or from the attached host (a packet of lower priority). The input registers store one flit and extract its routing information, [1].

Next element is DMUX_Host, which forwards the flit to the host if routing function signals that destination node is already reached. Else, if the above condition is not met, the flit is delivered to the relevant output port (though the next demultiplexer), depending again on the routing decision.

The outputs of buffers are connected to the output ports through a four non-blocking crossbars. The output ports are the common resource, which arbitration logic has to consider, when evacuating a flit from the buffers. They should be marked as occupied as soon the header

ISBN: 978-960-474-297-4

flit enters and freed when the corresponding tail flit exists.

The composition of header, payload and tail flits (forming one packet) makes a virtual path across the ports and the buffers, which could not be cut by other such virtual paths and the becomes blocked, till common resources are freed.

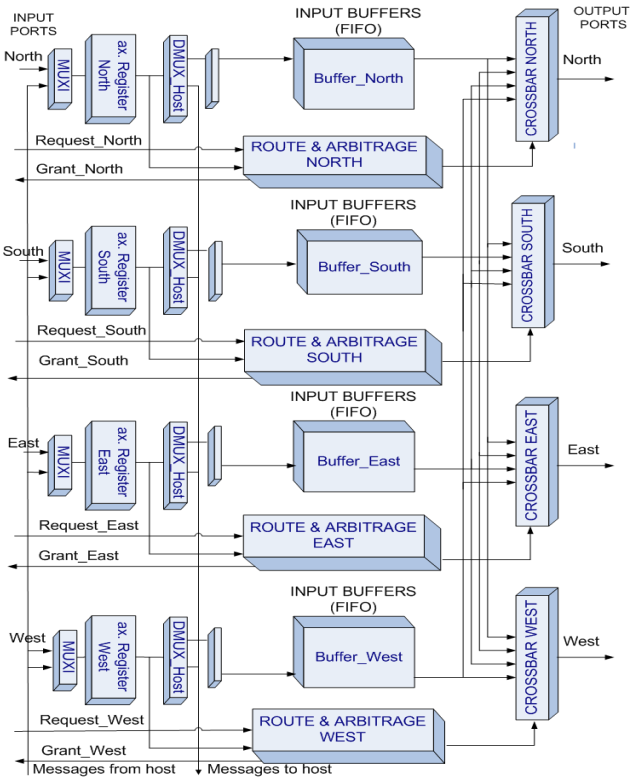

Fig. 1 High-Speed Switch Design

Suggested switch design uses low latency wormhole routing. This mechanism outperforms store and forward in terms of latency characteristics and at the same time requires smaller buffers because allocated buffer resources are at smaller units – flits.

The simulation model of proposed switch design is built using OMNET++. OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, with an Eclipse based-IDE and a graphical runtime environment.

Switch's model is subdivided into three stages: input registers, FIFO buffers and non-blocking crossbar, implemented as fully connected multiplexers and demultiplexers. They are represented as simple module, making the whole switch a compound module in OMNeT++'s model contexts. As it could be seen in Fig. 2, these simple modules have number of instances, corresponding to the number of ports, which is a configurable parameter in our model, allowing constructing of different topologies with any base and radix. All simple modules, building the switch are interconnected by channels as well as the links between switches themselves. Channels consist of data path and control links, which the editor (NED Editor) shows as just one line.

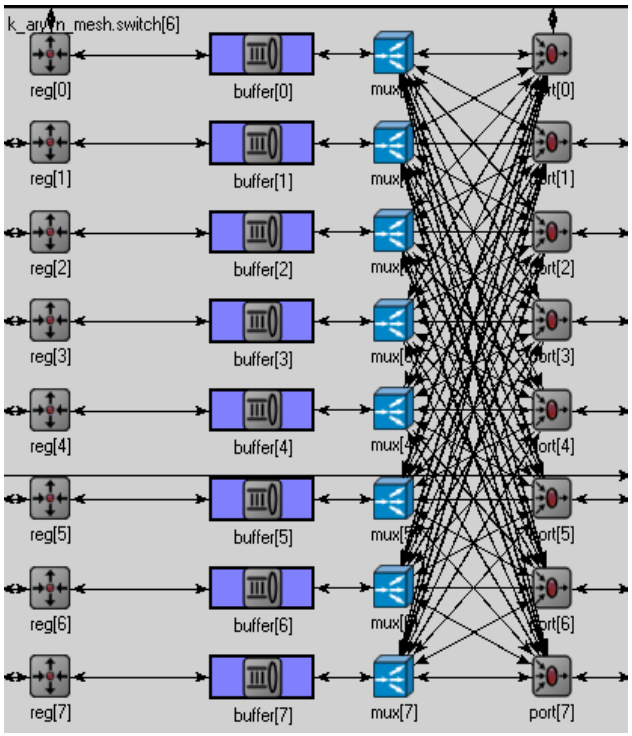

Fig. 2 Simulation View of Proposed Switch Design

The compound module: the switch has a global clock, which is called directly (without using massage mechanism) via Module::Clock () virtual function, which each module has. This allows easily defining the sequence of three stages pipelined clock for: input registers, buffers and output ports. The next step is to write modules behavior in C++. Each module defines its own functionality, but common strategy is: module keeps its input data/control state, when it receives a message and forwards it to the next module on the system clock (when its Clock() method has been invoked).

Host, which is a separate simple module, is the traffic generator and also traffic sink in our simulation model. It maintains its own clock, which is configured in each simulation experiment, giving us different values of the offered traffic rates.

ISBN: 978-960-474-297-4

3 TOFU System Area Network Design

TOFU – "torus fusion" is a six-dimensional mesh/torus topology which achieves highly scalable and fault-tolerant interconnection networks for large scale high-performance computer systems that can exceed 10 Petaflops [3]. It is developed by the Fujitsu engineers. Each network node consists of six links for **xyz** 3D Torus and four fixed sized links for **abc** 3D mesh/torus, thus forming a 6D mesh/torus from the Cartesian product of **xyz** and **abc**, Fig. 3.

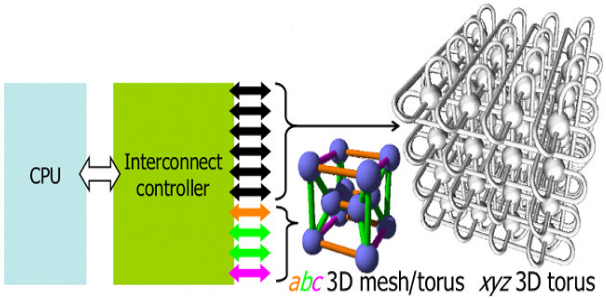

Fig. 3 TOFU Network Design

From another perspective, a node is an overlaid **xyz** torus that theoretically helps us to achieve twelve times higher scalability than the 3D torus network, Fig. 4.

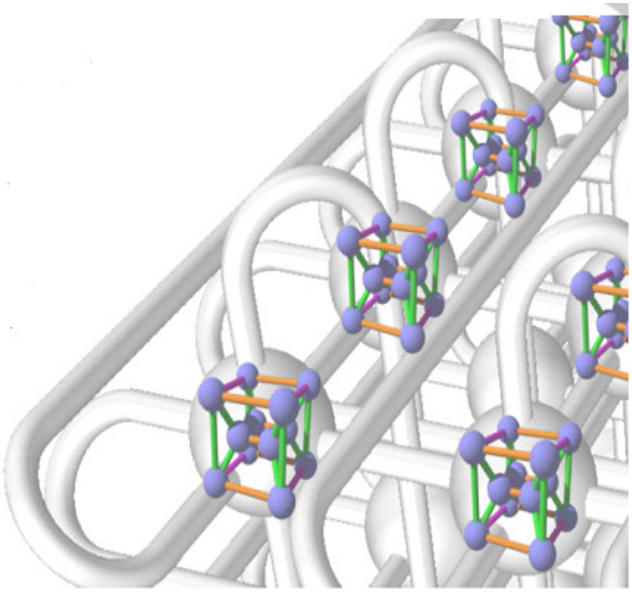

Fig. 4 TOFU Node Design

Each pair of adjacent **abc** mesh/torus is interconnected via twelve links, Fig. 5.

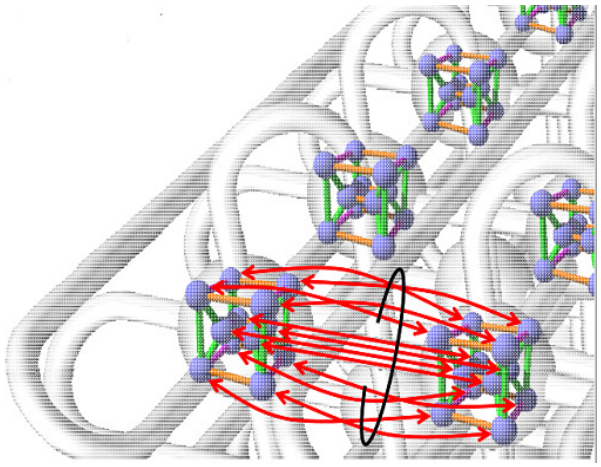

Fig. 5 TOFU Node Links

The routing algorithm implemented in TOFU network is path traversal first in the **abc** mesh/torus followed by **xyz** torus and then again in the **abc** mesh/torus (**abc=>xyz=>abc**), Fig. 6.

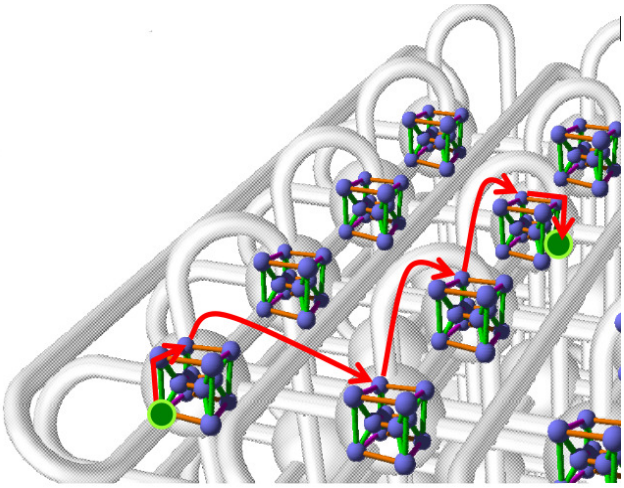

Fig. 6 TOFU Routing Algorithms

In case of faulty nodes, the multipath routing allows to detour faulty nodes in TOFU network, Fig. 7.

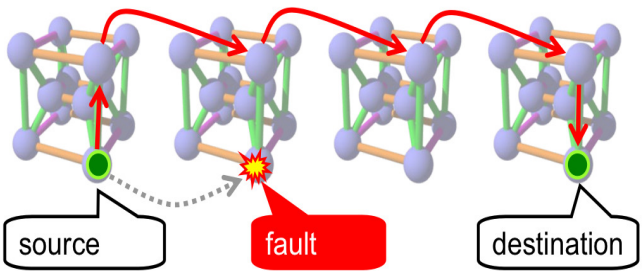

Fig. 6 Detouring Faulty TOFU Nodes

ISBN: 978-960-474-297-4

4 Experimental Framework

4.1 Platform

The developed parallel simulation models are tested and evaluated on IBM Blade Center with the hardware platform, based on three Blade servers, HS21, Xeon Quad Core E5405 80w 2.00GHz/ 1333MHz/12MB L2, 2x1GB Chk, two Blade servers, HS22, 2xXeon Quad Core E5504 80w 2.00GHz/ 800MHz/4MB L2, 3x2GB Chk, O / Bay SAS to disk subsystem IBM System Storage DS3400 Single Controller and hard disk drive subsystem for IBM 750GB Dual Port HS SATA HDD chassis specialist for Blade Center, IBM eServer BladeCenter (tm) H Chassis and recorder 2x2900W PSU UltraSlim, network switch Blade Center Chassis , BNT Layer 2 / 3 Copper Gb Ethernet Switch Module, Optical Switch chassis specialist for Blade Center, Brocade (R) 10-port 4 Gb SAN Switch Module with Optical Switch Module for IBM Short Wave SFP Module, together with the necessary wiring, special cabinet Blade Center, NetBAY S2 42U Standard Rack Cabinet and specialist-arrester Power Ultra Density Enterprise C19/C13 PDU Module (WW), located at the High-Performance and GRID Computer Lab, Computer Systems Department, Technical University of Sofia.

4.2 Simulation Tool

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, with an Eclipse based-IDE and a graphical runtime environment, [4]. The simulator provides efficient tools for users to describe the structure of the actual system. Some of the main features are: hierarchically nested modules; modules are instances of module types; modules communicate with messages through channels; flexible module parameters; topology description language.

The active modules are termed simple modules. They are written in C++ using the simulation class library. There are also extensions for real-time simulation, network emulation, alternative programming languages (Java, C#), database integration, System C integration.

OMNeT++ supports parallel simulation. The simulator is developed in a way that it encapsulates the simulation model with associated MPI code for parallel execution. The idea is, the developer to focus on creating the model and to simulate it in a parallel environment only with a few settings and adjustments.

4.3 Simulation Experiments and Result Analysis

The basic communication metrics include network latency and throughput. The network latency is defined as the average time for the massage transfer from the source node to the destination node. The network throughput is defined as the average number of packets, delivered for one machine cycle, or as the information in bits, transferred per unit of time.

The simulation experiments are targeted to evaluate the communication performance of the SAN architectural design of TOFU topology. The experiments imply different packet traffic - uniform and gossiping and variable of packet size – 32 and 64 flits. According to these input parameters the simulation experiments are conducted.

The TOFU system area network is tested for two traffic patterns – Uniform and Gossiping distribution in network.

Uniform Traffic: Each node sends messages to other nodes with an equal probability (i.e., destination nodes are chosen randomly using a uniform probability distribution function).

Gossiping Traffic: It is inspired by the form of gossip in social networks. Another name for gossiping is the so called “epidemic protocol”. The gossip traffic pattern spreads the information in a manner similar to the spread of a virus in a biological community – a host sends a message to a recipient. When the message is received, the recipient becomes a sender and sends messages to its neighbors and in such a way the problem grows recursively and the network load gets higher.

The packet count in all experiments is 100000. For both traffic patterns the tests are evaluated for package sizes of 32 flits and 64 flits.

For the simulation purpose, we choose a high-speed switch architecture detailed in second paper session, [1]. Though we changed the routing algorithm logic since we have a different topology. First of all we simplified the node representation from 6D to 2D.

ISBN: 978-960-474-297-4

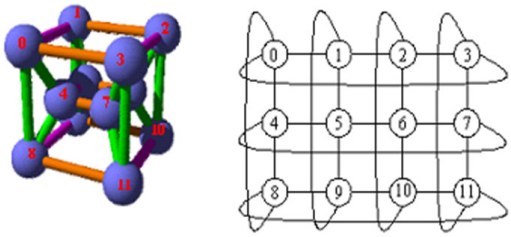

Fig. 7 Simulation Network Node

Secondly, we created three nodes with the design rules of Tofu. In our simulation each node knows its x and y coodinates. Therefore we easily know that numbers between 0 and 11 for 1 node, numbers between 12 and 23 another and so on. The routing algorithm reads the desired address from the flit head and first of all compares it according to the x coordinate if the x value of the desired address is the same with that of the switch the flit head starts moving to reach its y coordinate.

Network latency of 100 000 package count for uniform and gossip traffic patterns with package size 32 flits are shown in Fig. 8.

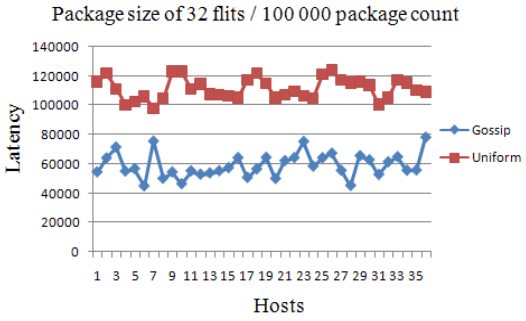

Fig. 8 Latency vs. hosts

Network latency of 100 000 package count for uniform and gossip traffic patterns with package size 64 flits are shown in Fig. 9.

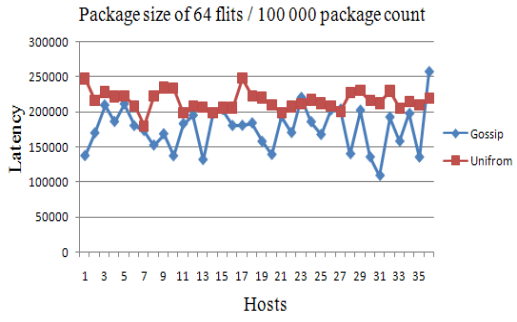

Fig. 9 Latency vs. hosts

The simulation results present better latency value of gossiping traffic distribution than the achieved results of uniform traffic distribution in network. The reason for those results is: in uniform traffic distribution the destination is with various distances, in many cases uniform destination can be close to the maximum distance that is possible in network. On the other site, in gossiping traffic distribution many of the nodes send messages to neighbor nodes. In this case the gossip route is closer to minimum distance nodes in network, respectively the time to reach the destination is less and the latency results are better.

ACKNOWLEDGEMENTS

The results reported in this paper are part of a research project ДЦВП 02/1, supported by the National Science Fund, Bulgarian Ministry of Education and Science.

References:

[1] Borovska, P. (2009), *Computer systems.* Sofia; Bulgaria: Ciela, ISBN 954-649-633-2 (in Bulgarian)

[2] Duato, J., Yalamanchili, S., Lionel M., (2002) *Interconnection networks: an engineering approach.* Morgan Kaufmann Publishers, ISBN 1-55860-852-4

[3] Ajima, Y., Sumimoto, S., & Shimizu, (2009). Tofu*: A 6D Mesh/Torus Interconnect for Exascale Computers*, Computer, Vol. 42 (11), pp. 36-40

[4] Varga, A., OMNeT++ version 4.0 User Manual, http://omnetpp.org

[5] Pl. Borovska, O. Nakov , D. Ivanova, K. Ivanov, G. Georgiev, *Communication Performance Evaluation and Analysis of a Mesh System Area Network for High Performance Computers*, 12-th WSEAS International Conference on Mathematical Methods, Computational Techniques and Intelligence Systems (MAMECTIS - 2010), Kantaoui, Sousse, Tunisia, May 3-6, 2010, ISBN: 978-960-474-188-5, pp. 217-222