Efficient Communication for FPGA Clusters

Stewart Denholm, Kuen Hung Tsoi, Peter Pietzuch, and Wayne Luk

Department of Computing, Imperial College London, UK
{swd10,khtsoi,prp,wl}@doc.ic.ac.uk

Abstract. Efficient communication between nodes is critical for achieving high performance in a computer cluster. Based on a dedicated inter-accelerator network, we enhance this communication with advanced networking functions, such as broadcasting and priority routing. This work enables decoupling user applications from physical network implementations, improving overall communication efficiency and modularity. A performance model is introduced taking into account application and platform specific parameters. Experiments are performed for various network configurations and application patterns. The results show up to a 55% reduction of communication time when employing our approach.

1 Introduction

Modern High Performance Computing (HPC) systems include heterogeneous hardware accelerators such as Field Programmable Gate Arrays (FPGAs.) There is usually no direct connection between these hardware accelerators and the cluster network. Gigabit Ethernet and the InfiniBand [1] network are two commonly employed communication technologies in HPC systems, as suggested by the Top500 supercomputer list [2]. Their involvement introduces significant overhead due to the data movement between the accelerators and the host memory.

To address this, HPC systems with heterogeneous accelerators are starting to incorporate direct inter-accelerator communications through dedicated connections. Figure 1 illustrates the idea of this inter-accelerator networking; the long, thick arrow shows a point-to-point (P2P) link. Similar configurations can be seen in existing FPGA based HPC systems [3, 6, 5]. Previous experiments indicate this helps improve the performance and scalability in applications [4].

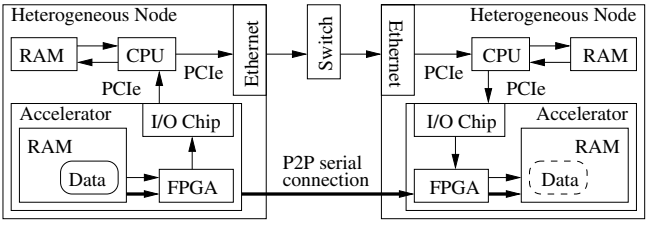


Fig. 1. Data transfer between accelerators in a heterogeneous cluster

O.C.S. Choy et al. (Eds.): ARC 2012, LNCS 7199, pp. 335–341, 2012.

The main objective of this work is to provide an efficient inter-FPGA communication framework without the need for centralised switching facilities. This can be achieved by supporting concurrent multi-destination addressing and by reducing redundant network traffic. The challenge lies in application-specific customisations, requiring the use of low level networking hardware. A parametrised user interface must be provided, as well as a model predicting performance gains.

The major contributions of this work include:

- A framework to support broadcasting and priority routing in an inter-FPGA P2P network. The data format, connection schemes, routing algorithms and architecture parameters can be customised for specific applications.
- A performance model for exploring the effects of various networking features and configurations. It provides an upper bound for expected performance and guides the decision making in each development stage by considering the customisable parameters.
- Experiments designed for measuring and evaluating the communication performance of the broadcast design. Results show an overall performance improvement of up to 55%, agreeing with the performance model estimation.

Various heterogeneous clusters equipped with FPGA-based accelerators have been reported in the last few years. The following presents several examples, and describes the differences between our work and that of others.

In the *sprit* cluster [6], there are 64 Xilinx ML410 evaluation boards connected via SATA cables, each channel of which provides up to 2.5Gbps bandwidth. The PowerPCs in the Virtex-4 FPGAs are used as the main controllers in the cluster. Previous experiments compare communication performance against physical factors, but it is unclear how the network is utilised at the application level.

The Axel cluster [4] utilises a 2D torus network via direct inter-FPGA connections. Networking functions are implemented in the FPGA fabric, avoiding software overhead in previous contributions. However, this work lacks advanced networking functions, such as broadcasting and priority routing.

The Maxeler platform includes a 10Gbps interface. Based on the original MaxRing technology [5], a CH2 connectivity expansion card is used to provide multiple channels with 10Gbps bandwidth. With this card, multiple nodes can be connected to form a larger ring, but other topolgies are not supported. Vendor specific APIs are used to control the data flow in the user application.

2 Broadcast Design

Without broadcasting, the user must send the same packet to each destination. This raises many issues. First, additional logic is needed to implement the multiple transmissions. Second, redundant packets occupy routing and queueing resources on all intermediate nodes. Third, addressing all the FPGAs in an application requires knowing the transmission path to each one; embedding this information in the application reduces the flexibility and scalability of the design.

Our broadcasting function addresses these issues without adversely affecting application performance. When encountering a broadcast packet, a router automatically duplicates it into the queues of all available outputs, storing a local

copy if the packet originated at another node. Previously seen packets are discarded automatically. Following this cascading transmission method, packets are delivered to all participating accelerators regardless of the connection topology.

The implementation and performance evaluation of the framework is based on the network shown in Figure 2, but can be adapted to other topologies without any low level modifications. Each of the 16 compute nodes contain an AXM-XRC-5T2 board from Alpha-Data, interfaced to the host system through the PCIe bus. As suggested by Figure 1, the PCIe bus and Ethernet do not provide an efficient path for inter-FPGA communication. In the proposed framework, the RocketIO GTP serial communication resources in the Xilinx Virtex-5 FPGA are utilised as the main data communication channel. With the XRM-HSSDC2A I/O connectivity card, each FPGA accelerator is capable of sending and receiving data through four independent ports in full-duplex. Single channel InfiniBand cables, rated for 2.5Gbps, connect the accelerators to form a 2D torus. This hardware configuration is similar to the cluster in [3].

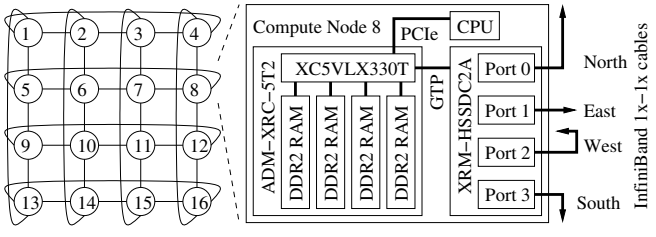


Fig. 2. A 2D torus network utilising P2P FPGA serial communication

3 Performance Model

Table 1 lists the parameters captured by our performance model. The platform specific parameters can be obtained from actual measurement and profiling on the target platform, or from the data specifications of the hardware components. The application specific parameters are obtained by extracting and abstracting the communication pattern of the target algorithm.

For a 1-to-all communication, the data from a single node must be sent to all other nodes to complete the process. The simplest model for the time T to send a packet from one node to another is:

$$T_{pkt} = p/B_l, \quad T_L = T_l + T_r + T_{pkt}, \quad T = (k-1) \times T_L + 2 \times T_a$$

where T_{pkt} is the time for transmitting a packet out of a hub, and T_L is the time for the packet to pass through a link. In a broadcasting scenario, the maximum number of links, $\hat{k}$, is used since that path will take the longest time to transmit. The value is dependent on the application and network topology. In our 16-node, 2D torus network, $\hat{k} = 4$ when the application is broadcasting to all nodes. The above analysis is modified as follows to represent 1-to-all communication:

$$T_P = \lceil d/p \rceil \times (T_L + T_a) + (\hat{k} - 1) \times T_L + T_a \tag{1}$$

Table 1. Parameters for network performance modelling

symbol	unit	meaning	typical value
platform parameters			
p	bits	size of a packet	16 - 528
q		max. packets in a queue	64
ρ		error rate in link transmission	5%
B_l	Mbps	bandwidth of physical link	1600
T_l	μs	FPGA link transmission latency	0
T_r	μs	FPGA packet routing latency	0.57
T_a	μs	application to routing logic latency	0.13
application parameters			
d	bits	total data size to be transferred	N/A
λ	s^{-1}	average packet rate	N/A
k		number of links along a path	N/A
$\tilde{k}$		average links per path	N/A
$\hat{k}$		maximum links per path	N/A

When a unicast packet is created, it will, on average, occupy $\tilde{k}$ number of slots in the queues along the path. For the duration of T', there will be $T' \times \lambda$ unicast packets created. For a cluster with N FPGA nodes and each node with four output queues, the network loading can be approximated as:

$$T_q = T_{pkt} \times (T' \times \lambda \times \tilde{k})/N/4 \tag{2}$$

When there is an error detected during packet transmission, the packet is automatically retransmitted within the link layer. Thus the effects of transmission errors in each node are independent and localised. The new link latency is:

$$T'_L = (T_L + T_q) \times (1 + \rho). \tag{3}$$

This new T'_L can be used in a more accurate approximation of the 1-to-all communication performance as:

$$T_{1-to-all} = \lceil d/p \rceil \times (T'_L + T_a) + (\hat{k} - 1) \times T'_L + T_a \tag{4}$$

4 Results

Our design is implemented within the CusComNet [4] framework on Xilinx Virtex-5 LX330T FPGAs. The throughput and latency of broadcast transmissions are measured and compared to CusComNet's unicast transmissions to multiple destinations. The InfiniBand cable line speed is set to 2Gbps due to observed instabilities above this value. Packet buffer sizes are set to 64 packets, each packet having a payload of 64 bytes. The 16-node cluster translates to address widths of 4 bits. The maximum number of packet identifiers is set to 8192 to ensure no two en-route packets will have the same identifier.

Our experiments are not designed to show the benefit of broadcast over multiple unicasts, but instead seek to determine the efficiency of the broadcast protocol, and whether the customisations have any detrimental effect.

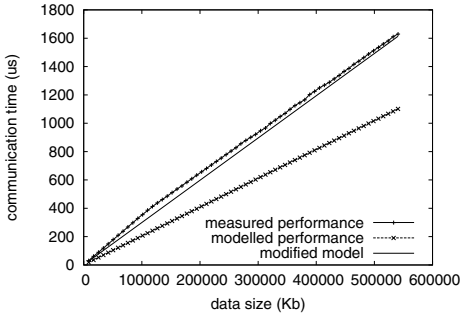
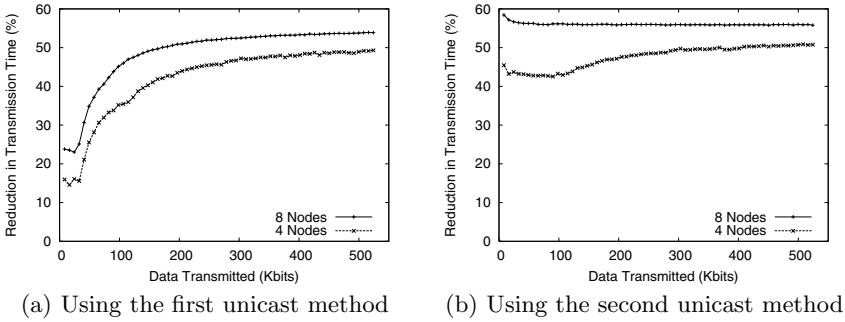


Fig. 3. Comparing measured and estimated performance in a 4-node network

Based on the model from Section 3, we measure the performance of 1-to-all broadcasting in a 4-node network comparing the communication times for various data sizes. $\hat{k}$ is set to 2 according to the network topology. The other platform specific parameters set to their typical values. Figure 3 compares the model to the observed results. The divergence may be due to the difference between the estimated T'_L and the actual link transmission time. Adding $0.5\mu s$ overhead to the T_L value, shown by the smooth line, matches the measured performance. We thus conclude, the model accurately represents of the system.

Figure 4(a) shows our implementation's improvement over multiple unicasts increases with data size, plateauing at around a 53% reduction in transmission time. As data sizes increase, packet buffers saturate and we can better compare the routing and packet handling of broadcast and unicast. Improvements are due to two factors: reduced buffer occupancy due to de-duplication of packets; and the use of the cascading packet routing algorithm. The latter provides benefit as the shortest path may not be the fastest due to packet congestion.

The question still remains as to whether these improvements are due to the efficient operation of the design or the inherent improvement of broadcast over multiple unicasts. Figure 4(b) shows the percentage speedup of our broadcast



(a) Using the first unicast method (b) Using the second unicast method

Fig. 4. Reduction in transmission time for broadcasting when implemented within CusComNet

implementation over a second unicast methodology, giving a similar result to Figure 4(a): about a $50-55\%$ speedup. The difference in speedup is most likely due to the first methodology's ability to utilise multiple output ports, thereby performing some operations in parallel and making use of additional buffer space.

When applied to different packet buffer depths in a 4-node cluster, Figure 5 shows our broadcast protocol's benefit is reduced when the packet buffers are not saturated. With larger data volumes, broadcast's reduction in total time versus multiple unicasts converges to around 50%. These results show our implementation does not have a negative impact on communication performance, and improves overall transmission times by $50-55\%$ for large data volumes.

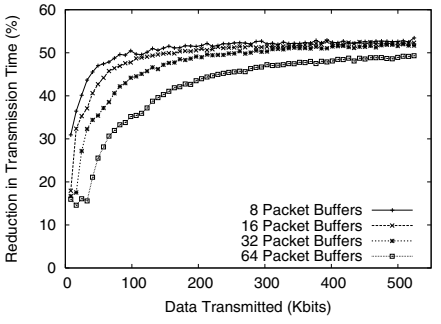


Fig. 5. Reduction in transmission time for broadcasting when using packet buffers of different sizes; using the first unicast methodology

5 Conclusion

This paper presents our customisable broadcasting framework. We outline the broadcast operation and develop a model to analyse its performance. Experiments show our broadcast implementation achieves up to a 55% reduction in transmission time when implemented within the CusComNet framework. We show our work benefits communications regardless of cluster size or packet buffer depths. We demonstrate a cascading broadcast routing algorithm that transmits data along all possible paths, and can therefore work with any network topology.

Acknowledgements. The support of Imperial College London Research Excellence Award, UK Engineering and Physical Sciences Research Council, Alpha Data, Maxeler, nVidia and Xilinx is gratefully acknowledged. The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement number 248976 and 257906.

References

1. InfiniBand architecture specification release 1.2.1. White Paper (2010)
2. TOP 500 supercomputer sites (2010), `http://www.top500.org/lists/2011/06`
3. Baxter, R., et al.: Maxwell - a 64 FPGA supercomputer. In: Proc. Conference on Adaptive Hardware and Systems (AHS), pp. 287–294 (2007)

4. Denholm, S., Tsoi, K.H., Pietzuch, P., Luk, W.: CusComNet: A customisable network for reconfigurable heterogeneous clusters. In: Proc. IEEE Int. Conf. on Application-specific Systems, Architectures and Processors, ASAP (2011)
5. Lindtjrn, O., Clapp, R.G., Pell, O., Mencer, O., Flynn, M.J.: Surviving the end of scaling of traditional micro processors in HPC. IEEE HOT CHIPS 22 (2010)
6. Sass, R., et al.: Reconfigurable computing cluster (RCC) project: Investigating the feasibility of FPGA-based petascale computing. In: Proc. IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM), pp. 127–140 (2007)