INTEGRATION, the VLSI journal xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Ultra-low latency communication channels for FPGA-based HPC cluster

Roberto Sanchez Correa, Jean Pierre David[*]

Department of Electrical and Computer Engineering, École Polytechnique de Montréal, Montréal, Canada

ARTICLE INFO

Keywords:
FPGA cluster
Interconnection network
Low latency
High-Performance Computing

ABSTRACT

The FPGA technology offers numerous advantages in terms of parallel computation, which is supported by on-chip low latency communications. Nevertheless, clustering FPGAs to achieve a larger computing power may require external high-speed and low-latency communication channels. Because of the overhead due to complex features and functionalities, existing off-the-shelf IP cores for high-speed standard communication often waste valuable clock cycles and bandwidth. This paper presents the implementation of an ultra-low latency inter-FPGAs communication IP suitable for high performance computing machines. Our IP achieved 272 ns (34 clock cycles) half-round trip end-to-end latency and an aggregate bandwidth of 16 Gbps per node on Virtex-5 FPGA. To test the proposed IP under a high-performance situation, we implemented an eight-FPGA parallel computing machine hosting 48 coprocessors interconnected through our custom designed network. Experimental results show a global computational efficiency of 97.6%. The proposed architecture is scalable and easily portable to most recent FPGAs, which should lower the latency and increase the bandwidth even more.

1. Introduction

Over the last decade, the High-Performance Computing (HPC) community has been interested in FPGA technology for acceleration at small scale (e.g. task or algorithms with FPGA development boards), and at large scale (e.g. complex systems with powerful FPGA-based clusters).

The growing markets of data centers and cloud computing leverage the expertise from the HPC community to find solutions to the ever-increasing demand for high-performance. New compute-intensive workloads such as Deep Neural Networks (DNN) for Artificial Intelligences (AIs), machine learning, Big Data analytics, and 4K video processing, necessitate powerful heterogeneous servers to meet the demand.

Many companies have fast response time as a top-of-mind business requirement. This imposes hard-constraints to the execution and data propagation times across the data center. Such latency-sensitive workloads come mostly from:

- *Telecom companies*. With the Centralized/Cloud Radio Access Network (C-RAN) architecture [1] gaining more attention from the wireless infrastructure industry, the telecom companies demand real-time cloud computing, low and deterministic latency operation, flexible hardware accelerators and high-speed switching performance to achieve the centralization of all base stations' computational resources and virtualize the functionalities of the baseband units. The 5G vision challenges even lower latency requirements with down to 1 ms end-to-end application time [2].
- *Financial institutions*. The High-Frequency Trading (HFT) market [3,4] provides a critical competitive advantage to those traders with the highest trade execution rates. “A 1 ms advantage in HFT applications can be worth $100 million a year to a major brokerage firm” [5]. Hence, the domain of algorithmic trading and market data-feeds demands real-time HPC to run intense computing algorithm, calculate risk and take decisions within the shortest possible delay.
- *Media and Web companies*. With millions of users streaming, sharing, broadcasting and consuming media content across the internet, the data centers must move, store, encrypt, search and analyze petabytes of data in real-time. With a user waiting for the response of a requested service in an acceptable delay, the system-wide response time must be minimized. Both compute and network latencies play a critical role in the overall performance perceived by the front-end user.

Cloud providers like Microsoft and Amazon have relied on FPGAs for the emerging next generation of cloud server and to reduce the gap between the current performance demand and the current performance supply [6,7]. By integrating a FPGA into each server, highly intense computing applications are offloaded from the CPU and accelerated onto the FPGA. Amazon has gone even further by facilitating FPGA cloud services to the entire world with the Amazon Elastic Compute Cloud (Amazon EC2) F1 instances [8]. This new cloud infrastructure

[*] Corresponding author.
E-mail addresses: roberto.sanchez-correa@polymtl.ca (R. Sanchez Correa), jpdavid@polymtl.ca (J.P. David).

https://doi.org/10.1016/j.vlsi.2018.05.005
Received 29 June 2017; Received in revised form 9 April 2018; Accepted 11 May 2018

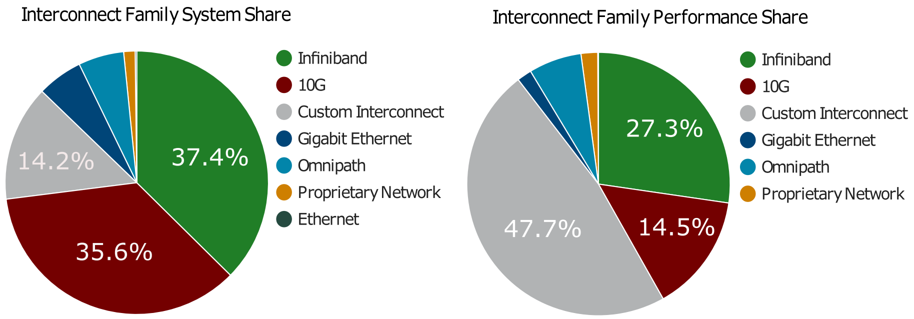

Fig. 1. HPC Interconnect families (November, 2016) [9].

provides an internet accessible FPGA-based accelerator pool (cluster) that could be used for HPC. Nevertheless, are these new enhanced servers squeezing the best of the FPGA technology for clustering?

In any HPC system, the computational units and the interconnect network are the two main components. The performance capabilities of the HPC is compromised, among others, by the network performance in terms of latency, bandwidth, fault tolerance, and scalability. Most data centers and supercomputers around the world use standard interconnections to communicate the computational units [9]. However, because of the overhead due to complex features and functionalities, existing high-speed standard communication protocols often waste valuable clock cycles and bandwidth. Custom interconnections, on the other hand, are the third most utilized interconnect family for supercomputers and the first interconnect type in terms of performance when comparing to standard and proprietary solutions (Fig. 1). Furthermore, for those HPC system involving FPGA clusters, it has been demonstrated that the standard commercial-off-the-shelf interconnects for FPGA-to-FPGA communication lack performance to support time-critical applications [10]. Instead, custom communication approaches are preferred.

We propose a custom lightweight high-speed and low latency communication IP suitable for real-time latency-sensitive HPC applications on multi-FPGA platforms. Our architecture utilizes a parameterizable number of hardware sockets to allow and simplify network accesses from both hardware and software environments. By means of a custom packet-based protocol and well-selected functionalities, our approach reduces the latency penalty associated with high-speed complex communication protocols, reaching one of the lowest in the literature. As a result, tightly-coupled design partitions across FPGA-based clusters are possible. To demonstrate it, we implemented a large dense Matrix-Vector Multiplication (MVM) acceleration kernel with 48 computational units distributed among eight FPGAs.

The MVM is the core of many applications, including neural networks, which are of recent interest for low latency Artificial Intelligence services like Google's Assistant and Apple's Siri. More work would be necessary to implement a complete neural network application. However, the purpose of the MVM is, firstly, to validate the functionality of the proposed architecture and, secondly, to show the potential of our low latency communications on a mainstream algorithm.

Our IP facilitates the network accesses to 48 units in a cluster. Because of the low latency transfers, the experimental tests showed that our tightly-coupled FPGA-based MVM kernel excels over others, even if in many cases, the MVM kernels are embedded in single-chips or loosely-coupled clusters where the computational units communicate through an on-chip network or don't communicate at all. Furthermore, we show the potential system performance penalty associated with wrong choices of communication protocols.

The remainder of this paper is organized as follows: Section 2 reviews the work related to this research. Section 3 presents the proposed communication IP architecture. Section 4 presents a multi-FPGA interconnection network and an HPC test utilizing the communication IP as the Network Interface Controller. Section 5 concludes this work.

2. Related work

We have categorized the related works into two main groups: pure multi-FPGA machines and FPGA-based network accelerators on heterogeneous machines. A third subsection presents some MVM implementations on reconfigurable hardware for performance comparison in the result section.

2.1. Pure multi-FPGA machines

The *Spirit* cluster in Refs. [11–13] was built of 64 Xilinx ML-410 development boards hosting Virtex-4 FPGAs. This implementation uses Xilinx's Aurora protocol to interface 8 MGTs along with a custom SATA breakout board, providing up to eight connectivity channels per node at a bit rate of 3.2 Gbps. Such a high radix (8 connectivity channels) has the advantage of better network topology exploration hence smaller network diameter. However, higher FPGA resources were used due to the on-chip packet switching among the channels. Another drawback is the I/O bounds related to the switch implementation, which limits the bandwidth to 3.2 Gbps per channel even with channel bonding. The inter-nodes latency reported is 0.8 µs.

A quad-FPGA cluster implementation on a Berkeley Emulation Engine 3 (BEE3) multi-FPGA prototyping platform is presented in Ref. [14]. At the Physical and Link level of the OSI model, they used Xilinx's Ten-Gigabit Attachment Unit Interface (XAUI) and Ten-Gigabit Ethernet Media Access Controller (10GbEMAC) IP pair. The system only leverages one high-speed serial port (used as the system entry) out of eight available, bounding the platform aggregate bandwidth to only 10 Gbps. As expected, the high overhead and complexity of internet protocol implementation resulted in around 45% FPGA resources utilization. Even if network interface latency was not reported, we can deduce it from the physical and link layer implementations, that it is above 1 µs. A similar Internet protocol acceleration approach has been presented in Ref. [15] where a full TCP/IP stack was implemented, reporting 5.5 µs.as the lowest stack latency.

In Ref. [16], eight BEE3s are interconnected to build a 32-FPGA cluster. Two serial protocols are implemented: 10 Gigabit Ethernet and Aurora. The 10GbE approach reported 840 ns channel latency and 1.56 µs network diameter. The Aurora implementation was made possible thanks to CX4-to-SATA breakout cables, which extend platform connectivity beyond the eight CX4 ports. The switch implementation on such high connectivity configuration adds a significant delay to the packet routing. As a consequence, the system shows almost no improvement in the network diameter (845 ns). In terms of Bit Error Rate, this approach does not implement any error detection/correction mechanism. Nevertheless, in order to reduce channel faults, the transceivers are forced to operate at a line rate of 1.95 Gbps instead of the initial 3.75 Gbps. The lowest FPGA resources utilization and the channel latency reported are around 10%, and 541 ns respectively.

Bluehive in Refs. [10,17,18] is a custom FPGA-based cluster composed of 64 DE4 boards hosting Altera's Stratix IV chips. Each board has 12 SATA3 channels, achieved with a PCIe-to-SATA breakout expansion