Towards Low-Latency Communication on FPGA Clusters with 3D FFT Case Study

Jiayi Sheng Chen Yang Martin C. Herbordt
Department of Electrical and Computer Engineering
Boston University, Boston, MA

***Abstract*—FPGA-based clusters are excellent candidates for future High Performance Computing systems as they allow for the integration of communication and computation in the same device. This integration, however, requires tight coupling which in turn means that clock jitter among FPGA boards and the variance of link latencies can constrain the performance and increase the difficulty of synchronization among the nodes. In this paper we investigate the link latency variance and clock jitters of two high-end GiDEL/Altera FPGA boards. We find that, while surprisingly noticeable, these effects are manageable with standard IP. We apply these findings to a real-life case study, the 3D-FFT, and are able to refine our implementation. This is now more than twice as fast as our previously published result and 20x faster than the fastest published result on a CPU cluster.**

***Keywords*-FPGA; Low Latency Communication; Multi-Gigabit Transceiver; High Performance Computing; FFT**

I. Introduction

A critical limiting factor in high performance computing (HPC) performance is scalability: as processor capability continues to increase through ever higher core counts and use of accelerators, the bottleneck becomes communication. To address this problem we are investigating use of direct and programmable communication (DPC). *Direct* in that communication proceeds directly from accelerator to accelerator; *programmable* in that application-awareness can be taken into account within the router switch. Our testbed is the Novo-G# FPGA cluster, an extension of the Novo-G [1]. Several FPGA clusters have been built with direct interconnects [2]–[5]; our goal here is to investigate their utility for general-purpose HPC. As a case study we are investigating one of the hardest applications for which to obtain strong scaling: long time-scale molecular dynamics and the relatively small 3D FFTs that are a critical part of that application [6]–[10].

FPGA clusters are appropriate for DPC for several reasons. CPUs and GPUs currently do not support direct interconnect while ASIC clusters like Anton2 [11] are necessarily application specific. Other attributes of FPGA clusters are as follows. First, the FFT (and other) IP is highly efficient [12], [13]. Second, the Multi-Gigabit Transceivers enable low-latency and high-bandwidth FPGA-to-FPGA connections [14], [15]. And third, the intrinsic flexibility and reusability of FPGA offers acceptable development cost when compared with ASICs.

This work was supported in part by the National Science Foundation through Award #CNS-1405695 and by the Charles Stark Draper Laboratory through their URAD Program Award #SC001-0000000834. Email: (jysheng|cyang|herbordt)@bu.edu

When implementing applications on a single FPGA, the design is often easy to synchronize. If we want to expand the design to a large-scale cluster this issue is more problematic. There is no global clock for all the nodes in one cluster. Therefore, two identical computation kernels on two different nodes will generally not finish simultaneously. For communication, the latency on different physical links will differ as well: tranceiver clock rates vary and cables have different lengths. The way to deal with these variances, or jitter, is through adding complexity to the communication mechanism such as increased buffer sizes and throttling through flow control. These add overhead. What we investigate here is the nature of this overhead and how it affects application performance.

II. Background

Our target architecture is FPGA clusters with direct FPGA-FPGA communication of the kind described below. Specifics are from the Novo-G#, a 128-node 3D torus extension to the Novo-G cluster [1]. The transceiver is based on the Interlaken-PHY IP core of the Altera Stratix-V FPGA family [16]. We use the Interlaken-PHY IP core to implement the direct FPGA-FPGA communication. The Interlaken PHY IP core is a lightweight, scalable, high-bandwidth and low-overhead FPGA IP family supporting both Altera and Xilinx FPGA devices.

In general the source of latency variations in SerDes devices resides in both serial and parallel parts [17]. On the serial side is a frequency multiplier from the input clock of the transmitter to the output clock of the serial link. On the receiver side a low-frequency clock is recovered from the high-frequency signal on the serial link with a frequency divider. Potential phase change happens during the frequency multiplication and division. Also, the phase change is non-deterministic, which implies one source of the latency variation of the data transfer on the link. In addition, the cable lengths and the serial clock frequencies vary from link to link. On the parallel side, the elastic FIFOs introduce another source of latency variation. This is because latencies in the FIFOs are determined by the distance between the read and write points assuming the reading rates and writing rates are the same. The positions of these pointers, however, is affected by many factors; these can be approximated as random if no dedicated mechanism is introduced.

Most 3D FFT implmentations are built from 1D and 2D FFTs (see e.g., [18]). The 3D FFT can be viewed as data manipulation on a 3D array, which is essentially 1D FFTs calculated on all

three dimensions. Since the 3D FFT has N^3 data points, each of the three dimensions requires N^2 N-point 1D FFTs for a total of $3N^2$ 1D FFT calculations. The 1D FFTs are first computed in the x dimension, then the y, and finally on the z dimension. Each dimension has to wait for the previous dimension to finish before it can start.

III. Platform

A. Hardware

Our test fixture is a server with dual CPUs (for control only) and four high-end Gidel FPGA boards. The CPUs are Intel Xeon E5-2620 v2 processors @2.10 GHz, each with 12 physical threads. The four FPGA boards are ProceV D8 mother boards each with a 3QSPF daughter board [19]. The boards are connected to the host via PCIe buses. The FPGA is the Altera Stratix V 5SGSMD8K2, which has high capacity (952K LEs), and memory and I/O performance. The FPGAs have 26 12.5/14.1 Gb/s Transceivers (MGTs) that can utilize PCIe, CXP, and SFP+ interfaces. The aggregate bandwidth is up to 366 Gb/s (full duplex). All of these make the ProceV ideal for low-latency, high performance networking and HPC applications.

B. Software

We used the Quartus II v14.0 suite, Gidel ProcWizard v9.2, and related tools to manage the design flow. Gidel ProcWizard generates Quartus II project files as well as the pin assignment files. Then we used the Altera MegaCore generator to generate IPs, including transceivers, and the Qsys system integration tool to instantiate the transceiver entities and finish the interconnection among transceivers and control logic. Next, we used ModelSim to simulate the design and Quartus II v14.0 to compile, synthesis, and place-and-route. After programming the FPGAs, the Transceiver Toolkit was used to monitor the bit error rate (BER) of the transceiver links and to adjust the parameters of the Physical Medium Attachment (PMA). The signalTap tool was used to probe the waveforms of the signals of the designs for debugging the design.

IV. Latency and Jitter Measurements

In an FPGA cluster there two major forms of jitter: clock jitter among different nodes and variations of latencies among different links. We examine these in turn and produce a simple model to be used as the basis for accurate simulations.

A. Variation of Communication Latencies

The latency variations reside at four levels:

1) in one lane over time,
2) among different lanes of the same link,
3) among different links of the same FPGA chip, and
4) among different transceivers of the different FPGAs.

We performed experiments to obtain results for all four.

We use the classic ping-pong test and validated by examining the waveforms directly. The FPGA on the right has a traffic generator which sends data to the MGT interface in a parallel format. At the same time, the monitor creates and records a random value for the outbound data stream of the transmitter. As data is transferred to the tx parallel port module, a counter in the monitor is initiated. The data is then sent through the serial link to the FPGA on the left side, after which the data is routed from that FPGA's rx port to its tx port and transmitted back again. The FPGA on the right side eventually receives the return data and a comparator checks it with the recorded sent data. Once that happens, the counter in the monitor block stops. This mechanism is very accurate, working on the frequency from one PLL.

We deployed two FPGAs to determine the variation among the boards. For each board, two MGTs are configured with Interlaken PHY IP to explore the variation among the MGTs on the same board. Each MGT is configured with 4 lanes to discover the variations among the lanes of the same link table. The physical characteristics for the Interlaken PHY IP are shown in Table I.

To explore level-one latency variation (single lane) we sample its latency 75 times and plot the results (see Figure 1). We find that distribution appears to be random with distribution TBD. The mean and variance are shown in Table II. The level-two variations are found by comparing the different lanes in the same transceiver. The level-three variations exist in the differences between the two MGTs of the same FPGA. The level-four variations can be identified by comparing the differences between FPGA0 and FPGA1. When we look at the variation over time for a single lane, the maximum latency is about 40% larger than the minimum (on average). In some extreme cases (Lane3 of the MGT1 on FPGA0), the maximum latency is close to double of the minimum latency.

When we compare the difference among multiple lanes and boards, however, the variation is relatively trivial compared to the fluctuation in the timing dimension.

TABLE I
Physical Characteristics of the Interlaken IP

Attribute	number	Attribute	number
metaframe length	2048	Data rate	10Gbps
Number of Lanes	4	ALMs	240
AlMs used for memory	0	Combinational ALUTs	394
Dedicated Logic Register	427	I/O Registers	0
Block Memory Bits	0	DSP Blocks	0

TABLE II
Mean and variance of link latencies. All times in ns.

FPGA0	MGT0				MGT1			
Lane	0	1	2	3	0	1	2	3
Avg	179.5	175.4	177.1	179.8	175.5	175.4	173.3	175.3
Min	140	156	96	160	152	144	144	136
Max	204	196	200	208	200	204	204	200
STD	12.6	11.3	16.9	11.5	11.3	11.5	9.9	12.7
FPGA1	MGT0				MGT1			
Lane	0	1	2	3	0	1	2	3
Avg	179.1	177.6	176.7	178.2	175.7	177.1	175.6	175.1
Min	140	156	156	152	140	116	152	136
Max	208	200	208	200	204	204	204	200
STD	14.3	11.4	11.2	11.5	13.7	12.8	11.8	11.8

Figure 2 gives us a different perspective with the (a) panel giving latency distributions for different MGTs and the (b) panel different FPGAs. While not a perfect fit, we use a Gaussian with mean of 176.8ns and Standard Deviation of 12.4 ns to improve

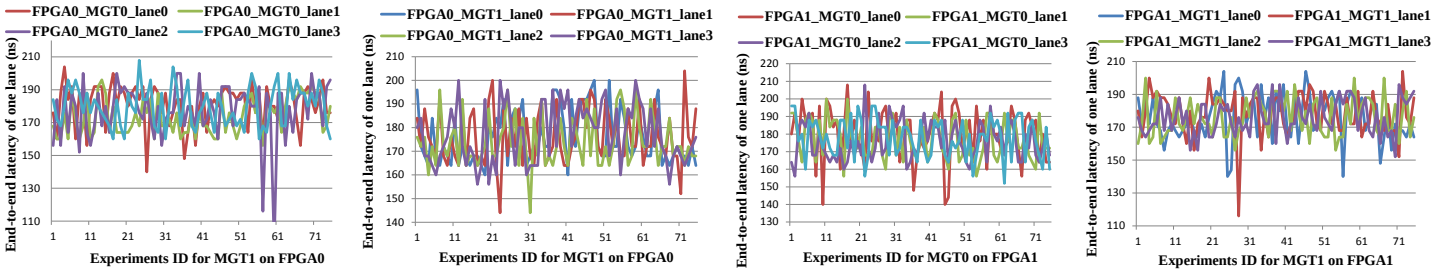

Fig. 1. Variations of end-to-end latency over time for four lanes

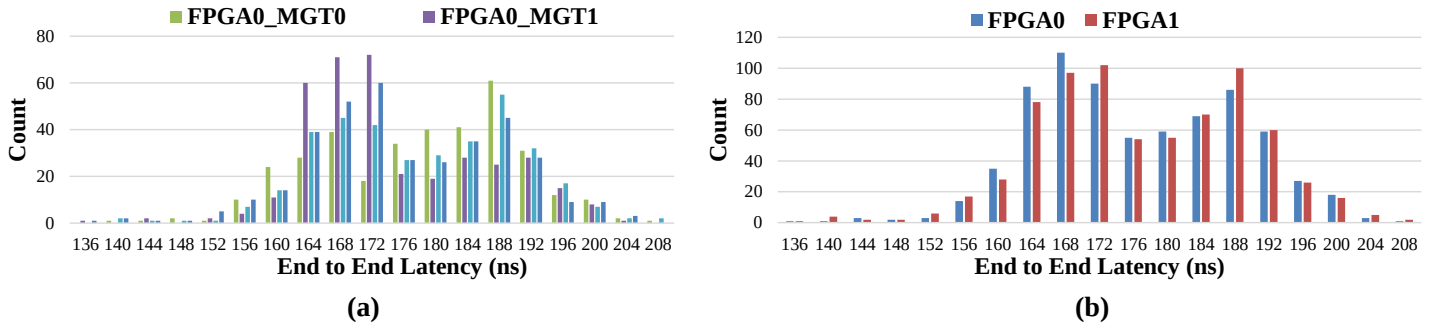

Fig. 2. (a) Latency distribution comparison among four Multi-Gigabit Transceivers (MGTs) (b) Latency distribution comparison between two Altera Stratix V boards

on the fixed time estimate used previously to simulate the 3D FFT.

B. Measurements of Clock Jitter

The clocks on the different boards do not have perfectly matched clock frequencies. To measure the jitter we use the output clock of a PLL at 250 MHz as a reference to measure the frequency of clk0 signals of the four ProceV boards.

To make the experiment concrete, on the four boards we instantiate a single 16-point 1D FFT IP. We run this module 1024 times on each and measure the execution time with respect to the reference clock. The measured effective frequencies are shown in Table III. The difference between the longest and shortest cycle is 6 ps. While this result appears to be trivial, it is actually significant for practical applications. For example, for a 256^3 FFT the number of clock cycles spent on the computation will be around 1000. Then the jitter between two neighboring nodes could be as large as 6 ns, which is in the same order of magnitude as the variation in link latency.

TABLE III
CLOCK JITTER OVER FOUR PROCEV BOARDS

	FPGA0	FPGA1	FPGA2	FPGA3
freq(MHz)	103.174	103.213	103.239	103.200
cycle(ns)	9.692	9.689	9.686	9.690
jitter (ps)	6	3	0	4

V. CASE STUDY: 3D FFT

In the previous section, by conducting several simple experiments, we obtained an estimate of link latency, variation of link latency, and clock jitter. In this section we apply this information to the simulation of the 3D FFT.

A. Simulated Network Architecture

As described in our previous work [7], our FPGA-based cluster topology is a 3D-torus. The routing scheme is table-based routing [20], [21]. Table-based routing means that there are one or more routing tables in each node, and there is an corresponding entry in each routing table for each incoming packet. The packet header carries index information to find correct entry. After the correct entry is found, the information in the entry is used to determine the next node for the packet. For the network switch architecture we use a simple ring-based design (also as described in [7]). The ring is composed of 7 routers, six of which deal with the traffic on the torus links and one which injects or ejects packets to or from the network. If there is no congestion, each packet spends just one cycle on each (internal) router. In general, each packet spends 2-4 cycles on each node.

B. Support for Application-Aware Routing

The 3D FFT is a well-studied application. Our previous work [7] proposes a generalized mapping for 3D FFTs of arbitrary size 3D-tori with arbitrary numbers of nodes. The mapping defines the source and destination of each packet. By knowing these *a priori* we can use table-based routing to orchestrate the packet routes so as to minimize congestion. We note that the current communication architecture allows sufficiently deterministic packet arrivals to support this form of application-aware routing.

C. Results

In the previous work [7], our simulations assumed fixed link latency and did not take clock jitter into account. In this paper, we replace fixed link latency with the varied latencies that obey the Gaussian distribution, and also add the clock jitter into

the simulation. Table IV demonstrates the new FFT simulation results and the comparison with the results that use fixed latency. Cluster size is 4^3, FPGA operating frequency is 150MHz, and the data type is 32-bit floating point. The right three columns have latencies in microseconds. "Fixed" denotes average latency with no variance, "Variable" denotes that variance has been modeled, and "Ref" denotes previous results where we assumed (very conservatively) 100MHz FPGA frequency and 500ns fixed link latency [7].

TABLE IV
3D FFT LATENCY CASE STUDY.

fft size	fixed	variable	reference
16^3	1.63	1.64	3.98
32^3	2.24	2.48	5.46
64^3	6.72	6.92	16.76
128^3	35.13	41.14	101.11

We note that despite the despite the apparently substantial variance, the two new results are not that different. This is mainly because the variation of the latency on the physical links and the clock jitter are small when compared with the latency caused by congestion. But still, the variation of physical links and clock jitter degrade the performance for all cases because they increase the cost for synchronization.

VI. Discussion and Future Work

We have investigated variance in communication latency at various levels and clock jitter among FPGAs. We are not aware of other studies on these topics. We find that these quantities are significant and in the case of link variance, surprisingly large. We also find, however, that the communication infrastructure based on the Altera MGTs and the Interlaken PHY link IP is sufficient to make these variances and jitter manageable.

The fact that there is substantial jitter has at least two effects: there must be flow control between communication and application modules (already in place in our case study) and the link protocol must have appropriate buffer sizes and flow control mechanisms. This second effect means that the communication IP necessarily cannot be too light-weight. This has two effects: chip resources and latency. As we see from Table I the chip resources needed are negligible. Latency, however, through additional buffering and flow control, may be increased by nearly a factor of two over a lighter weight alternative, say, Lowlatency Interlaken.

On the positive side, the current setup gives seamless communication; only simple flow control and no additional buffering are needed on the application side. This indicates the viability of certain modes of higher-level packet processing which assume a synchronous flow model. These include off-line routing and computation within the network.

The 3D FFT case study shows more than 2x performance increase over our previous results [7]. The key point here is that for a fixed-size non-trivial cluster (in this case 64 nodes), the computation is harder, not easier, than for smaller problem sizes. For example, the fastest published 3D FFT on a CPU cluster is 2ms for a problem size of 128^3 on a 512 core BlueGene/Q [22]. This is only slightly faster than for a single CPU (2.4ms). Our result here is nearly $20\times$ better than that.

This manuscript represents work in progress. We are in the middle of conducting more extensive latency and jitter tests and are also examining other communication IP. More significantly, we hope to present a full working 3D FFT on the FPGA cluster at the conference.

References

[1] A. George, H. Lam, and G. Stitt, "Novo-G: At the Forefront of Scalable Reconfigurable Computing," *Computing in Science and Engineering*, vol. 13, no. 1, 2011.

[2] T. Bunker and S. Swanson, "Latency-Optimized Networks for Clustering FPGAs," in *Proc. IEEE Symp. on Field Programmable Custom Computing Machines*, 2013.

[3] S. Moore, P. Fox, A. Markettos, and A. Majumdar, "Bluehive–A Field Programmable Custom Computing Machine for Extreme-Scale Real-Time Neural Network Simulation," in *Proc. IEEE Symp. on Field Programmable Custom Computing Machines*, 2012.

[4] A. Putnam, et al., "A reconfigurable fabric for accelerating large-scale datacenter services," in *Proc. Int. Symp. on Computer Architecture*, 2014, pp. 13–24.

[5] A. Schmidt, S. Datta, A. Mendon, and R. Sass, "Investigation Into Scaling I/O Bound Streaming Applications Productively with an All-FPGA Cluster," *International Journal on Parallel Computing*, 2013.

[6] A. Lawande, A. George, and H. Lam, "Simulative analysis of a multidimensional torus-based reconfigurable cluster for molecular dynamics," in *Proc. Workshop on Heterogeneous and Unconventional Cluster Architectures and Applications*, 2014.

[7] J. Sheng, B. Humphries, H. Zhang, and M. Herbordt, "Design of 3D FFTs with FPGA clusters," in *Proc. High Performance Extreme Computing Conference (HPEC)*, 2014, p. TBD.

[8] M. Chiu and M. Herbordt, "Molecular dynamics simulations on high performance reconfigurable computing systems," *ACM Trans. Reconfigurable Tech. and Sys.*, vol. 3, no. 4, pp. 1–37, 2010.

[9] B. Sukhwani and M. Herbordt, "Acceleration of a Production Rigid Molecule Docking Code," in *Proc. IEEE Conf. on Field Programmable Logic and Applications*, 2008, pp. 341–346.

[10] T. VanCourt and M. Herbordt, "Rigid molecule docking: FPGA reconfiguration for alternative force laws," *Journal on Applied Signal Processing*, vol. v2006, pp. 1–10, 2006.

[11] D.E. Shaw, et al., "Anton 2: raising the bar for performance and programmability in a special-purpose molecular dynamics supercomputer," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2014, pp. 41–53.

[12] Altera. (2014) FFT MegaCore Function: User Guide. [Online]. Available: http://www.altera.com/ literature/ug/ug_fft.pdf

[13] Xilinx. (2014) LogiCORE IP Fast Fourier Transform v9.0: Product Guide for Vivado Design Suite. [Online]. Available: http://www.xilinx.com/ support/documentation/ ip_documentation/ xfft/v9_0/pg109-xfft.pdf

[14] Altera. (2014) Stratix V Device Handbook volume2: Transceivers. [Online]. Available: http://www.altera.com/ literature/hb/stratix-v/ stx5_xcvr.pdf

[15] Xilinx. (2009) Virtex-5 FPGA RocketIO GTP Transceiver. [Online]. Available: http://www.xilinx.com/ support/documentation/ user_guides/ug196.pdf

[16] Altera. (2015) Altera Transceiver PHY IP Core User Guide. [Online]. Available: http://www.altera.com/ literature/ug/ xcvr_user_guide.pdf

[17] R. Giordano and A. Aloisio, "Fixed-latency, multi-gigabit serial links with xilinx fpgas," *Nuclear Science, IEEE Transactions on*, vol. 58, no. 1, pp. 194–201, 2011.

[18] B. Humphries, H. Zhang, J. Sheng, R. Landaverde, and M. Herbordt, "3D FFTs on a Single FPGA," in *Proc. IEEE Symp. on Field Programmable Custom Computing Machines (FCCM)*, 2014, pp. 68–71.

[19] Gidel. (2013) ProceV Preliminary Product Brief. [Online]. Available: http://www.gidel.com/ pdf%5CProceV%20Product%20Brief.pdf

[20] W. Dally, P. Carvey, and L. Dennison, "The Avici Terabit Switch/Router," in *Proc. Sixth Symp. Hot Interconnects*, 1998, pp. 41–50.

[21] M. Kinsy, M. Cho, K. Shim, and M. lis, "Optimal and Heuristic Application-Aware Oblivious Routing," *IEEE Transactions on Computers*, vol. 62, no. 1, pp. 59–73, 2013.

[22] M. Guarrasi, G. Erbacci, and A. Emerson, "Autotuning of the FFTW library for massively parallel supercomputers," PRACE: Partnership Advanced Computing Europe, Tech. Rep., 2014.