

# Design of 3D FFTs with FPGA Clusters

Jiayi Sheng Ben Humphries Hansen Zhang Martin C. Herbordt

Department of Electrical and Computer Engineering  
Boston University; Boston, MA 02215

## ABSTRACT

The three dimensional Fast Fourier Transform (3D FFT) is widely applied in various scientific applications. Distributed 3D FFTs require global communication: this becomes a serious concern when strong scaling is required as in long timescale molecular dynamics simulations. In this paper, we propose a parameterized 3D FFT design that targets at a 3D-torus FPGA-based network of various sizes. Characteristics include direct FPGA-FPGA communication links, support for various internal switch designs, and use of table-based routing which saves chip area and routing cycles. We find that even assuming extremely conservative parameters, we are able to run the  $16^3$  FFT in  $3.9\mu\text{s}$ ,  $32^3$  FFT in  $5.46\mu\text{s}$ ,  $64^3$  FFT in  $9.52\mu\text{s}$ , and  $128^3$  FFT in  $25.72\mu\text{s}$ . These results indicate that clusters based on commodity FPGAs are likely to be appropriate when strong scaling is needed in applications limited by the 3D FFT.

Keywords—3D FFT, FPGA, High Performance Computing, Low-Latency Communication

## 1. INTRODUCTION

The three dimensional Fast Fourier Transform (3D FFT) is essential to numerous applications in diverse domains. In Molecular Dynamics (MD) simulations the 3D FFT reduces the complexity of computing the long range interactions. In molecular docking, the 3D FFT computes the scores for ranking the different conformations of molecular complexes [13, 23]. In imaging the 3D FFT accelerates algorithms that decrease scan time [14].

Especially interesting to this work is when the 3D FFT is both on the critical path and operating in a fixed sized problem domain, i.e., when strong scaling is needed. An example is MD simulations of biomolecules. These often have from a few 10s of thousands to a few 100s of thousands of particles and need to execute for E+9 to E+15 timesteps (fs) and beyond. The non-FFT part of the computation scales well and takes roughly 1s per timestep per CPU core. With a 1K core cluster, the non-FFT MD simulation of a protein takes about 10 days for  $1\mu\text{s}$  simulated time (E+9 timesteps). To get into the ms range requires, e.g., 100K cores and 100 days. The problem is that, as the cluster size increases, and while the problem size remains fixed, the compute time per timestep necessarily decreases. In these examples they are 1ms and  $10\mu\text{s}$ , respectively.

These calculations define the time budget for the 3D FFT in protein simulation: preferably in the  $\mu\text{s}$  range of compute time for FFT sizes of from  $16^3$  to  $128^3$  [22, 28]. Given the communication latencies of commodity networks, achieving these numbers poses a substantial challenge. The two ways to address the problem are to reduce the number of communicating nodes, e.g., by using accelerators, and to reduce the communication latency. In conventional clusters these are generally in conflict: accelerators must traverse extra hops in order to cooperate. This problem was solved by DE Shaw by building a dedicated ASIC-based computer, Anton [21].

FPGAs are an alternative. There are several reasons why they

are appropriate. First, they are commodity parts. Second they are ideally suited for both FFT [2, 10, 27] and non-FFT [4, 5, 6] parts of MD computations. Third, FPGAs' Multi-Gigabit Transceivers enable low-latency and high-bandwidth FPGA-to-FPGA connections [3, 25]. And finally, the intrinsic flexibility and reusability of FPGAs offers an acceptable development cost when compared with ASICs.

In this paper, we propose and evaluate scalable 3D FFT designs with a 3D-torus FPGA-based network. Our contributions as follows.

1. By fully utilizing the device-specific features on the FPGA, such as Multi-gigabit transceivers (MGT) and high-speed FFT IPs, our 3D-FFT design is able to achieve the same order of magnitude of latencies as Anton [28], and in some cases surpass them.
2. We create a framework to generalize the communication patterns for 3D FFTs with arbitrary number of points and for 3D-torus networks of arbitrary size.
3. We investigate two kinds of switch architectures, ring and crossbar, and compare their resource utilization and impact on overall performance.
4. We use table-based routing to route the packets. All the routing paths for each packet are precomputed offline and stored in routing tables. This saves both chip area for routing logic and routing cycles.
5. We have created a full-system cycle-accurate simulator that can simulate not only 3D FFTs, but also other applications on real FPGA-based clusters. Parameters include FPGA chip resources and FPGA-to-FPGA link latency and bandwidth.

The rest of the paper is organized as follows. In Section II the algorithms of the 1D and 3D FFTs are reviewed. In Section III the network architecture used in our design is described; this involves network topology, table-based routing, and switch architectures (ring and crossbar). In the next section, the implementation details are presented including the framework to generalize the communication pattern for the 3D FFT. We then show the experimental results and comparison with other designs. Finally, we make our conclusions and briefly describe future work.

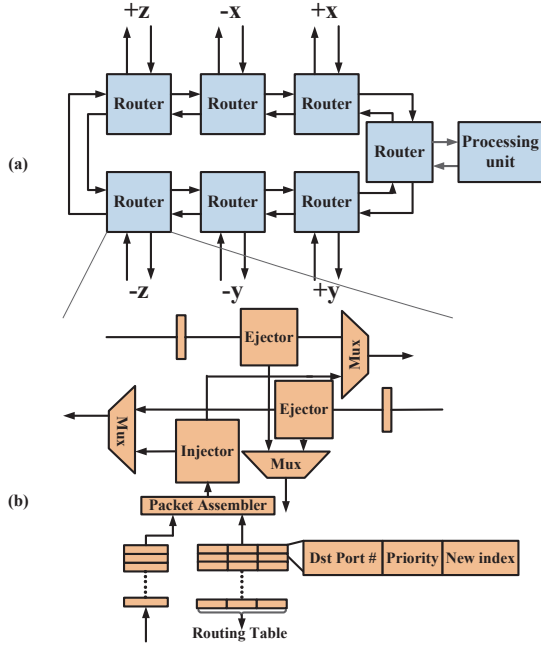
## 2. 3D FFT OVERVIEW

The 3D FFT data can be viewed as a cube of points, where each point represents a point of data in an FFT calculation. An  $N^3$  point 3D FFT can be expressed with Equation 1.

$$F(k_x, k_y, k_z) = \sum_{z=0}^{N-1} \sum_{y=0}^{N-1} \sum_{x=0}^{N-1} f(x, y, z) W_N^{xk_x} W_N^{yk_y} W_N^{zk_z} \quad (1)$$

where  $W_N = e^{-i\frac{2\pi}{N}}$ .





**Figure 2: (a) 7 node ring topology and (b) router microarchitecture for on-chip ring topology**

tion. Thus each input can get to its destination output in 1 cycle as long as there is no congestion.

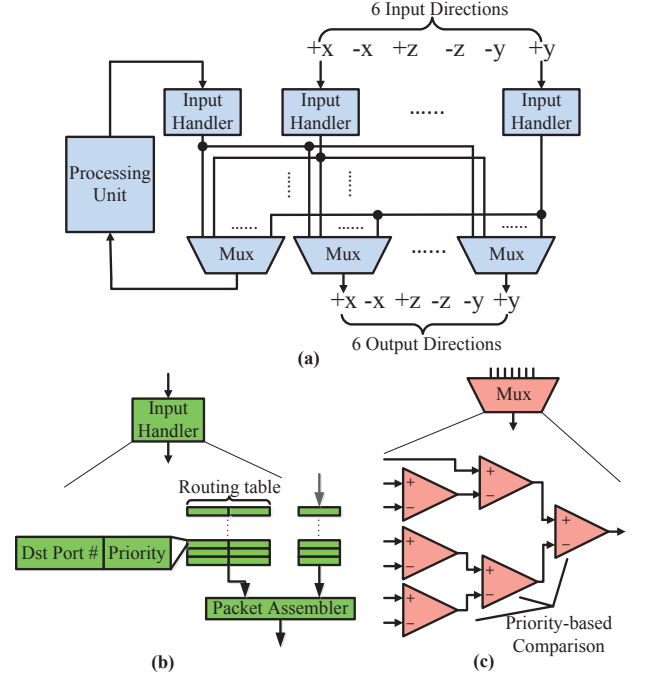
#### 4. IMPLEMENTATION

In this section we describe (i) the framework to generalize the communication pattern for the 3D FFT, (ii) the problem decomposition and data mapping, and (iii) a formula to estimate the latencies for various 3D FFTs ranging from  $16^3$  to  $128^3$ .

As described in the background section, the  $N^3$  3D-FFT can be decomposed into three phases, one per dimension. Each phase involves  $N^2$   $N$ -point 1D-FFTs. Between each pair of phases, there exists a communication phase to transpose the whole 3D array to get the data ready for the 1D-FFT on the next dimension. Following [28], we refer to the communication phase between  $X$  and  $Y$  as the  $XY$  corner turn and between  $Y$  and  $Z$  as the  $YZ$  corner turn. In [28], the authors summarize the communication pattern of  $XY$  corner turn and  $YZ$  corner turn for two specific cases:  $32^3$  FFT and  $64^3$  FFT on 3D-torus network with  $8^3$  nodes. In our work, we extend these two patterns to a general case of  $2^n \times 2^n \times 2^n$  3D FFT on 3D torus network of  $2^m \times 2^m \times 2^m$  nodes (see Figure 4).

In Figure 4, the first two lines show the most straightforward mapping of  $N^3$  data points onto an  $M^3$  torus cube, where  $N = 2^n$  and  $M = 2^m$ . There are then two cases for the permutation patterns:  $n < 2m$  and  $n \geq 2m$ . Both of them require that  $3m \leq 2n$ , which guarantees that there is at least one 1D FFT on one node. This is because there are  $2^{3m}$  nodes in total and  $2^{2n}$  1D FFTs on each dimension, which means there are  $2^{2n-3m}$  1D FFTs per node.

In Figure 4, there is a communication phase called  $X$ -fold, which takes place before computing the  $X$  dimension FFTs. This step transposes data from the initial mapping to get ready for this computation. In our design, this step is done offline, so the  $X$ -fold phase in Figure 4 does not cost any clock cycles. The permutations for the  $XY$  and  $YZ$  corner turns are also shown in Figure 4. Each expression determines the exact location of each datum on the network and has five overscored terms. The first three terms (in the



**Figure 3: (a) The network switch with crossbar architecture, (b) Inner structure of input handler, and (c) details in each mux**

parentheses) are binary expressions of the coordinates of the node that the data belongs to. The fourth term denotes the index of the  $2^{2n-3m}$  1D FFTs in that node. The fifth term denotes the index of the  $2^n$  points of the 1D FFT input data on that node.

We now illustrate how the permutations work ( $XY$  corner turn and  $YZ$  corner turns). Let  $n = 6$  and  $m = 3$ .

Binary expression for data (11,47,19)

$$\begin{aligned} &= \overline{x_5 x_4 x_3 x_2 x_1 x_0}, \overline{y_5 y_4 y_3 y_2 y_1 y_0}, \overline{z_5 z_4 z_3 z_2 z_1 z_0} \\ &= (001011)_2, (101111)_2, (010011)_2 \\ &= 11, 47, 19 \end{aligned}$$

Initial Mapping:

$$\begin{aligned} &(\overline{x_5 x_4 x_3}, \overline{y_5 y_4 y_3}, \overline{z_5 z_4 z_3}) \\ &= (001)_2, (101)_2, (010)_2 \\ &= 1, 5, 2 \end{aligned}$$

After  $X$  fold:

$$\begin{aligned} &(\overline{z_2 z_1 z_0}, \overline{y_5 y_4 y_3}, \overline{z_5 z_4 z_3}), \overline{y_2 y_1 y_0}, \overline{x_5 x_4 x_3 x_2 x_1 x_0} \\ &= ((011)_2, (101)_2, (010)_2), (111)_2, (001011)_2 \\ &= (3, 5, 2), 7, 11 \end{aligned}$$

After  $XY$  corner turn:

$$\begin{aligned} &(\overline{z_2 z_1 z_0}, \overline{x_5 x_4 x_3}, \overline{z_5 z_4 z_3}), \overline{x_2 x_1 x_0}, \overline{y_5 y_4 y_3 y_2 y_1 y_0} \\ &= ((011)_2, (001)_2, (010)_2), (011)_2, (101111)_2 \\ &= (3, 1, 2), 3, 47 \end{aligned}$$

After  $YZ$  corner turn:

$$(\overline{y_2 y_1 y_0}, \overline{x_5 x_4 x_3}, \overline{y_5 y_4 y_3}), \overline{x_2 x_1 x_0}, \overline{z_5 z_4 z_3 z_2 z_1 z_0}$$

Original data (3D array):

$$\overline{X_{n-1}X_{n-2} \cdots X_1X_0, Y_{n-1}Y_{n-2} \cdots Y_1Y_0, Z_{n-1}Z_{n-2} \cdots Z_1Z_0}$$

Initial Mapping:

$$\overline{(X_{n-1}X_{n-2} \cdots X_{n-m}, Y_{n-1}Y_{n-2} \cdots Y_{n-m}, Z_{n-1}Z_{n-2} \cdots Z_{n-m}), X_{n-m-1} \cdots X_1X_0, Y_{n-m-1} \cdots Y_1Y_0, Z_{n-m-1} \cdots Z_1Z_0}$$

When  $n < 2m$ :

After X-fold:

$$\overline{(Z_{n-m-1} \cdots Z_1Z_0Y_{n-m-1} \cdots Y_{2n-3m}, Y_{n-1}Y_{n-2} \cdots Y_{n-m}, Z_{n-1}Z_{n-2} \cdots Z_{n-m}), Y_{2n-3m-1} \cdots Y_1Y_0, X_{n-1} \cdots X_1X_0}$$

After X compute and XY corner turn:

$$\overline{(Z_{n-m-1} \cdots Z_1Z_0X_{n-m-1} \cdots X_{2n-3m}, X_{n-1}X_{n-2} \cdots X_{n-m}, Z_{n-1}Z_{n-2} \cdots Z_{n-m}), X_{2n-3m-1} \cdots X_1X_0, Y_{n-1} \cdots Y_1Y_0}$$

After Y compute and YZ corner turn:

$$\overline{(Y_{n-m-1} \cdots Y_1Y_0X_{n-m-1} \cdots X_{2n-3m}, X_{n-1}X_{n-2} \cdots X_{n-m}, Y_{n-1}Y_{n-2} \cdots Y_{n-m}), X_{2n-3m-1} \cdots X_1X_0, Z_{n-1} \cdots Z_1Z_0}$$

When  $n \geq 2m$ :

After X-fold:

$$\overline{(Z_{n-m-1} \cdots Z_{n-2m+1}Z_{n-2m}, Y_{n-1} \cdots Y_{n-m+1}Y_{n-m}, Z_{n-1} \cdots Z_{n-m+1}Z_{n-m}), Z_{n-2m-1} \cdots Z_1Z_0Y_{n-m-1} \cdots Y_1Y_0, X_{n-1} \cdots X_1X_0}$$

After X compute and XY corner turn:

$$\overline{(Z_{n-m-1} \cdots Z_{n-2m+1}Z_{n-2m}, X_{n-1} \cdots X_{n-m+1}X_{n-m}, Z_{n-1} \cdots Z_{n-m+1}Z_{n-m}), Z_{n-2m-1} \cdots Z_1Z_0X_{n-m-1} \cdots X_1X_0, Y_{n-1} \cdots Y_1Y_0}$$

After Y compute and YZ corner turn:

$$\overline{(Y_{n-m-1} \cdots Y_{n-2m+1}Y_{n-2m}, X_{n-1} \cdots X_{n-m+1}X_{n-m}, Y_{n-1} \cdots Y_{n-m+1}Y_{n-m}), Y_{n-2m-1} \cdots Y_1Y_0X_{n-m-1} \cdots X_1X_0, Z_{n-1} \cdots Z_1Z_0}$$

**Figure 4: Generalized mapping  $2^n \times 2^n \times 2^n$  3D FFT problem on  $3D 2^m \times 2^m \times 2^m$  torus network And data permutation pattern during two communication phases (XY corner turn and YZ corner turn).**

$$\begin{aligned} &= ((111)_2, (001)_2, (101)_2, (011)_2, (010011)_2) \\ &= (7, 1, 5), 3, 19 \end{aligned}$$

This example shows data movements when mapping a  $64^3$  3D FFT onto an  $8^3$  torus. The original  $X$ ,  $Y$ , and  $Z$  indexes for this datum are 11, 47, and 19 respectively. It will be mapped to the node whose coordinates are (1,5,2). After the X-fold, the coordinate of the node becomes (3,5,2), and the data will be mapped to the 7th 1D FFT IP. The relative address of this data on this IP is 11. In our design, the X-fold is done offline, therefore physically the initial location of this data is the 11th slot on the 7th IP on Node(3,5,2). After the XY corner turn, the datum is sent to the 47th slot on the 3rd IP on the Node(3,1,2). Finally the YZ corner turn puts the data on the 19th data slot on the 3rd IP on the Node(7,1,5). Based on this generalized permutation pattern, we can derive the number of packets that will be transmitted during each phase and how large each packet will be. These are displayed in Table 1.

**Table 1: The number of packets to be sent in each communication phase and the size of the packets**

Turn	FFT Size	Torus Size	condition	packets/node	Data/packet
xy turn	$2^{3n}$	$2^{3m}$	$2m > n$	$2^{3m-n}$	$2^{2n-3m}$
xy turn	$2^{3n}$	$2^{3m}$	$2m \leq n$	$2^m$	$2^{2n-3m}$
yz turn	$2^{3n}$	$2^{3m}$	$2m > n$	$2^n$	$2^{2n-3m}$
yz turn	$2^{3n}$	$2^{3m}$	$2m \leq n$	$2^{2m}$	$2^{2n-3m}$

To get a better sense of what would be the best network size for any particular 3D FFT, we created a generalized formula to estimate the latency of the entire 3D FFT computation. As the entire process can be decomposed into computations and communications, we can look at these two parts separately.

The computation latency is straightforward and can be easily found in Altera documentation [2]. The communication on the 3D torus is more complicated: the break-down of the estimate of the communication latency is shown in Table 2. Since the data throughput is limited by the bandwidth of the MGTs, we need to account for the time to output all the data on one node. This is calculated using the total number of data points per node and the bandwidth

of the internode links. Based on the general case above, the number of points per node should be  $2^{3n-3m}$ . The delay on the links is another important factor of the overall latency and is calculated based on the length of the longest path in the generalized routing pattern described above.

**Table 2: Estimated latency of communication in microseconds for various problem and network sizes**

Turn	FFT Size	Torus Size	Conditions	Estimation Formula BW: Bandwidth, LD: Link Delay
xy turn	$2^{3n}$	$2^{3m}$	$2m > n$	$\frac{2^{3n-3m}}{BW} + (2^{2m-n-1} + 2^{m-1}) \times LD$
xy turn	$2^{3n}$	$2^{3m}$	$2m \leq n$	$\frac{2^{3n-3m}}{BW} + (2^{m-1}) \times LD$
yz turn	$2^{3n}$	$2^{3m}$	$2m > n$	$\frac{2^{3n-3m}}{BW} + (2^m) \times LD$
yz turn	$2^{3n}$	$2^{3m}$	$2m \leq n$	$\frac{2^{3n-3m}}{BW} + (2^m) \times LD$

## 5. EXPERIMENTAL RESULTS

We built a cycle-accurate simulator to gather experimental results for our design. The simulator is configured with practical parameters based on the capabilities of current FPGA devices. The torus size in this paper is restricted to  $8 \times 8 \times 8$  and  $4 \times 4 \times 4$ , both of which are practical configurations in state-of-the-art technologies [8]. Compared with the 3D mesh, the 3D torus has a shorter diameter, which is beneficial in reducing all-to-all communication latency. Each node in the torus contains an FPGA. Each FPGA has a network switch and a local processing unit (as illustrated in Figure 2 and Figure 3). Each processing unit contains a number of 1D FFT IPs; this number is determined by the FFT size and the chip area. The 1D FFT IPs we adopted in our design are generated using Altera FFT MegaCore [2]. Latency and the maximum number of IPs that fit on a high-end Altera FPGA are shown in Table 3. The Altera FFT IP has the best latency in streaming mode: for an  $N$  point 1D FFT, the latency cycles is  $N$  cycles. The first output datum is ready as soon as last input datum is read in. Besides the 3D FFT, this simulator is also capable of simulating any application for which the communication pattern is given.



**Table 3: Altera FFT MegaCore latency and max number of IPs could fit on an Altera Stratix V 5SGSMD8**

1D FFT Size	Latency (cycles)	DSP Blocks Used	Max # of IPs per node	Total # of DSP Blocks
16	16	8	245	1963
32	32	16	123	1963
64	64	16	123	1963
128	128	24	82	1963

The node-to-node connections in our design use the Multi-Gigabit Transceivers. Using the vendor-supplied light-weight low-overhead protocol, the end-to-end communication latency of MGT is around 20-40 clock cycles in current technology [1, 26]. In our model, we conservatively pick 50 cycles as the node-to-node latency value and an internal clock frequency of 100MHz (the IP itself runs at 250MHz or more). This means that the node-to-node communication latency is fixed at 500ns. For internode bandwidth, we assume 51.2 Gbps, which is a plausible configuration for a single link implemented with 2 MGTs [3]. We assume single-precision floating point throughout.

We compared our two switch architectures during synthesis and post place & route using Quartus; the PP&R mapping results are shown in Table 4. The ring-based router does not show much advantages in chip area; this is because we only have seven ports for each router type. With more ports, as would be required to support a higher dimensional network, the resource requirements of crossbar would soon outstrip those for the ring.

**Table 4: Switch architecture resource utilization on an Altera Stratix V 5SGSMD8**

Switch Architecture	# of Logic	# of Registers
Ring	2485	3568
Crossbar	3312	2672

The performance results gathered after simulations are displayed in Table 5. Problem sizes ranging from  $16^3$  to  $128^3$  were simulated. Note that the best network size crosses over from  $4^3$  to  $8^3$  between FFT sizes of  $32^3$  and  $64^3$ . There is no simulation of the  $16^3$  FFT for the  $8^3$  cluster because that would require fewer than 1 FFT per FPGA per dimension. The estimated latencies are also included in Table 5 for comparison. In fact, the estimates are quite close to the simulation results. Note that there is not much difference between crossbar and ring. This is mostly because the crossbar reduces the latency by only a few cycles per hop compared with the ring, while the link latency is 50 cycles per hop. The benefit from using the crossbar for these networks and data sizes is therefore trivial compared to the impact of the link latency. One thing to notice is that when the FFT size is large, the crossbar has worse performance than the ring. The reason is that when the FFT size gets large, according to the communication pattern, all of the traffic tends to go into one dimension, making most of the crosspoints underutilized. For FFT sizes of  $128^3$  (and larger), congestion becomes critical (see Table 1), but also allows for the table based scheme to show most benefit.

In Table 6, we compare our results with results from CPUs, GPUs, ASICs, and other FPGA implementations. Except for ASIC (Anton) all are for single sockets. We find that communication overhead does not overwhelm the calculation and that performance of our design is faster than performance on CPUs and GPUs by at least one order of magnitude (achieving strong scaling for the target applications). Also, the FPGA cluster performance is similar to

**Table 5: Latency in microseconds for various problems sizes and network sizes. Simulations assume 10ns cycle time.**

FFT Size	Net. Size	IPs per Node	Switch Arch.	Estimated Latency	Simulated Latency
$16^3$	$4^3$	4	Ring	3.67us	3.98us
$16^3$	$4^3$	4	Xbar	3.67us	3.86us
$32^3$	$4^3$	16	Ring	5.27us	5.46us
$32^3$	$4^3$	16	Xbar	5.27us	5.30us
$32^3$	$8^3$	2	ring	7.65us	8.44us
$32^3$	$8^3$	2	Xbar	7.65us	8.23us
$64^3$	$4^3$	32	Ring	16.76us	16.76us
$64^3$	$4^3$	32	Xbar	16.76us	15.50us
$64^3$	$8^3$	8	Ring	9.23us	9.52us
$64^3$	$8^3$	8	Xbar	9.23us	9.32us
$128^3$	$4^3$	64	Ring	88.79us	101.11us
$128^3$	$4^3$	64	Xbar	88.79us	106.75us
$128^3$	$8^3$	32	Ring	20.11us	25.72us
$128^3$	$8^3$	32	Xbar	20.11us	26.74us

that of Anton [28]. For the  $64^3$  FFT, the design presented here is faster than Anton [28] by about 30%.

## 6. RELATED WORK

We now review other work implementing 3D FFTs on FPGAs. In [24], the authors focus on accelerating the 3D FFT by redesigning 1D FFT IP using Hard Embedded Blocks (HEB). However, their 1D FFT IP is still slower than the Altera’s IP. The authors in [19] reduce the data transfer time of XY corner turn and YZ corner turn by using a runtime configuration method called Coarse Grain Reconfigurable Architecture (CGRA). However, their method does not scale well. When problem size increases, the running time increases exponentially. In [11], the authors provide an effective 3D FFT implementation on a single FPGA. Preliminary work by one of the authors [9] investigates some switching issues in more depth but does not account for congestion. We have extended that here.

## 7. CONCLUSION AND FUTURE WORK

In this paper we present a design of mapping 3D FFTs onto an FPGA-based cluster with a 3D torus, direct connections between FPGAs, and offline routing. Even with extremely conservative assumptions, we demonstrate strong scaling. The overall conclusion is to demonstrate the viability of FPGA clusters for long timescale MD simulations of even modest sized proteins.

There is much room for improvement. First, as the cluster gets large the use of the FPGA becomes sparse. This enables the use of much faster 1D FFT IPs (broadside) that would be limited only by the FPGAs’ internal bandwidth. Second, our design does not include any overlap of communication and computation. And third, in this paper we use strict XYZ routing. Our off-line (table-based) scheme, however, allows for routing that is nearly congestion free.

## 8. REFERENCES

- [1] Altera. SerialLite III Streaming MegaCore Function User Guide, 2013.
- [2] Altera. FFT MegaCore Function: User Guide, 2014.
- [3] Altera. Stratix V Device Handbook volume2: Transceivers, 2014.
- [4] Chiu, M., and Herboldt, M. Molecular dynamics simulations on high performance reconfigurable computing systems. *ACM Trans. on Reconfigurable Technology and Systems* 3, 4 (2010), 1–37.

**Table 6: Results for various technologies and problem sizes. Anton is fixed point, otherwise results are for single precision floating point. All times are in microseconds. Release date is from corporate announcements of availability in quantity. Stratix-V times are our simulation results.**

Implementation Technology									Performance in $\mu\text{s}$		
Tech.	Make	Model	Parallelism	Part #	Proc.	Freq.	Rel. Date	Code	$16^3$	$32^3$	$64^3$
<i>2008 era technology</i>											
CPU	Intel	Nehalem	4 cores	E5530	45nm	1.6GHz	2009/Q1	MKL	38	116	983
GPU	NVIDIA	Tesla	240 SPs	C1060	55nm	1.3GHz	2008/Q3	CUFFT	54	66	257
FPGA	Altera	Stratix-III	single FPGA	EP3ES260	65nm	0.22GHz	2008/Q2	report	4.5	DNFit	DNFit
ASIC	DE Shaw	Anton	512 PEs	—	90nm	0.8GHz	2008/Q3	report	Not Av.	4	13
<i>2012 era technology</i>											
CPU	Intel	Sandy Bridge	8 cores	E5-2680	32nm	2.7GHz	2012/Q1	MKL	22	55	288
GPU	NVIDIA	Kepler	2688 SPXs	Tesla K20c	28nm	0.73GHz	2012/Q4	CUFFT	25	29	92
FPGA	Xilinx	Virtex-7	single FPGA	XC7v2000	28nm	various	2012/Q2	report	3.6	21	216
FPGA	Altera	Stratix-V	64 or 512 FPGAs	5SGSMD8	28nm	100MHz	2012/Q3	here	3.86	5.30	9.32

- [5] Chiu, M., Herbordt, M., and Langhammer, M. Performance potential of molecular dynamics simulations on high performance reconfigurable computing systems. In *Proceedings High Performance Reconfigurable Technology and Applications* (2008).
- [6] Chiu, M., Khan, M., and Herbordt, M. Efficient calculation of pairwise nonbonded forces. In *Proc. IEEE Symp. on Field Programmable Custom Computing Machines* (2011).
- [7] Dally, W., Carvey, P., and Dennison, L. The Avici Terabit Switch/Router. In *Proc. Sixth Symp. Hot Interconnects* (1998), pp. 41–50.
- [8] George, A., Lam, H., and Stitt, G. Novo-G: At the Forefront of Scalable Reconfigurable Supercomputing. *Computing in Science & Engineering* 13, 1 (2011), 82–86.
- [9] Humphries, B. Using Offline Routing to Implement a Low Latency 3D FFT in a Multinode FPGA System. Master’s thesis, Department of Electrical and Computer Engineering, Boston University, 2013.
- [10] Humphries, B., Zhang, H., Sheng, J., Landaverde, R., and Herbordt, M. 3D FFT on a Single FPGA. In *Proc. IEEE Symp. on Field Programmable Custom Computing Machines* (2014).
- [11] Humphries, B., Zhang, H., Sheng, J., Landaverde, R., and herbordt, M. 3D FFTs on a Single FPGA. In *Proc. IEEE Symp. on Field Programmable Custom Computing Machines (FCCM)* (2014).
- [12] Intel. Intel Xeon Phi System Software Developer’s Guide, 2013.
- [13] Katchalski-Katzir, E., Shariv, I., Eisenstein, M., Friesem, A., Aflalo, C., and Vakser, I. Molecular surface recognition: Determination of geometric fit between proteins and their ligands by correlation techniques. *Proc. Nat. Acad. Sci.* 89 (1992), 2195–2199.
- [14] Kim, D., Trazsko, J., Smelyanskiy, M., Haider, C., Dubey, P., and Manduca, A. High-Performance 3D Compressive Sensing MRI Reconstruction Using Many-Core Architectures. *Journal of Biomedical Imaging* 2011, 2 (2011).
- [15] Kim, J., and Kim, H. Router Microarchitecture and Scalability of Ring Topology in On-Chip Networks. In *Proceedings of the 2nd Internatioal Workshop on Networks on Chip Architectures* (2009), pp. 5–10.
- [16] Kinsy, M., Cho, M., Shim, K., and lis, M. Optimal and Heuristic Application-Aware Oblivious Routing. *Computers, IEEE Transacions on* 62, 1 (2013), 59–73.
- [17] Leighton, F. T. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufman Publishers, San Mateo, CA, 1992.
- [18] Murali, S., Atienza, D., Benini, L., and Micheli, G. A Method for Routing Packets Across Multiple Paths in NoCs with In-Order Delivery and Fault-Tolerance Gaurantees. In *Proceedings of the 43rd annual Design Automation Conference* (2006), pp. 845–848.
- [19] Nidhi, U., Paul, K., Hemani, A., and Kumar, A. High Performance 3D-FFT Implementation. In *Proc. IEEE Symp. on Circuits and Systems* (2013), pp. 2227–2230.
- [20] Pham, D., Aipperspach, T., Boerstler, D., and et al. Overview of the architecture, circuit design, and physical implementation of a first-generation cell processor. *Solid-State Circuits, IEEE Journal of* 41, 1 (2006), 179–196.
- [21] Shaw, D., Deneroff, M., Smelyanskiy, M., Dror, R., and et al. Anton, a Special-Purpose machine for Molecular Dynamics Simulation. *Communications of ACM* 51, 7 (2008), 91–97.
- [22] Sukhwani, B., and Herbordt, M. FPGA Acceleration of Rigid Molecule Docking Codes. *IET Computers and Digital Techniques* 4, 3 (2010), 184–195.
- [23] VanCourt, T., and Herbordt, M. Rigid molecule docking: FPGA reconfiguration for alternative force laws. *Journal on Applied Signal Processing v2006* (2006), 1–10.
- [24] Varma, B., Paul, K., and BalaKrishnan, M. Accelerating 3D-FFT Using Hard Embedded Blocks in FPGAs. In *Proc. VLSI Design and 2013 12th International Conference on Embedded Systems (VLSID), 2013 26th International Conference on* (2013), pp. 92–97.
- [25] Xilinx. Virtex-5 FPGA RocketIO GTP Transceiver, 2009.
- [26] Xilinx. LogiCORE IP Aurora 64B/66B v7.1, 2012.
- [27] Xilinx. LogiCORE IP Fast Fourier Transform v9.0: Product Guide for Vivado Design Suite, 2014.
- [28] Young, C., Bank, J., Dror, R., Grossman, J., Salmon, J., and Shaw, D. A  $32 \times 32 \times 32$ , spatially distributed 3D FFT in four microseconds on Anton. In *Proceedings of the Conference on High Performance Computing Networking* (2009), pp. 1–11.