CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
*Concurrency Computat.: Pract. Exper.* 2016; **28**:2374–2393
Published online 25 June 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3565

SPECIAL ISSUE PAPER

# Novo-G#: a multidimensional torus-based reconfigurable cluster for molecular dynamics‡

Abhijeet G. Lawande*,†, Alan D. George and Herman Lam

*NSF Center for High-Performance Reconfigurable Computing (CHREC), Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA*

SUMMARY

Molecular dynamics (MD) is a large-scale, communication-intensive problem that has been the subject of high-performance computing research and acceleration for years. Not surprisingly, the most success in accelerating MD comes from specialized systems such as the Anton machine. In this paper, we describe Novo-G# (*novo-jee-sharp*), a multi-node reconfigurable system designed for the acceleration of communication-intensive scientific problems in general, and MD in particular. This system provides a high-bandwidth, low-latency 3D torus network to allow direct communication between kernels running on multiple field-programmable gate arrays. We also present a performance model for Novo-G# running the 3D Fast Fourier Transform (FFT) kernel that forms the core of MD simulations. We validate the model against published Anton performance data and through initial hardware experiments on Novo-G#. Finally, through simulation studies, we show that this system at scale performs better than specialized systems like Anton and outperforms established CPU-based clusters like Blue Gene/Q by an order of magnitude for the 3D FFT kernel, with greater flexibility and lower costs. Copyright © 2015 John Wiley & Sons, Ltd.

Received 22 February 2015; Accepted 24 April 2015

KEY WORDS: high-performance computing; modeling; computer simulation; reconfigurable architectures; field-programmable gate arrays

## 1. INTRODUCTION

The field of computational science deals with the development of mathematical models to depict natural processes and solve scientific problems. Unlike laboratory experimentation, computational science is limited by the models available and the resources required to run them. It has been successfully applied to fields such as bioinformatics, fluid dynamics, and quantum mechanics [2] to build and test computational models of a complexity that cannot easily be replicated in the laboratory. A common theme in the aforementioned areas is the N-body problem: simulation of the interaction between multiple bodies because of the forces between them. At the molecular level, this simulation is termed molecular dynamics (MD) and can be applied to areas of science as varied as biomolecular engineering, neurobiology, material science, and nanotechnology. While originally

*Correspondence to: Abhijeet G. Lawande, NSF CHREC Center, Room 317, Larson Hall, University of Florida, Gainesville, FL 32611, USA.

†E-mail: lawande@chrec.org

‡Extension of Conference Paper: An earlier version of this work featured a conceptual design of the Novo-G# network and a performance model based on Anton [1]. Since that publication, an initial deployment of 32 nodes of Novo-G# has been completed. In this paper, we present the architecture of the deployed system and the design of a three-layer network stack to support the 3D torus network. We also describe the hardware architecture of the 3D fast Fourier transform kernel on eight nodes. Data from this implementation are used to improve the specificity of the performance model presented earlier. In conjunction with other optimizations to the model, we can now present simulation runs and predictions up to $8 \times 8 \times 8$ system size, the size of the largest Anton machine.

restricted to use on general-purpose systems, a number of specialized systems have since been developed for MD, such as MDGRAPE [3], Anton [4, 5], and Anton2 [6].

Anton was developed in 2008 as a specialized system designed to accelerate MD simulation by several orders of magnitude and bring millisecond-scale simulations of tens of thousands of atoms within reach [7]. While Anton has achieved this goal, the system is only available in limited quantity, can only be used for MD simulations, and incurs a high non-recurring engineering cost. Our goal is to design a system that can perform as well as Anton on problems like MD, while retaining the ability to accelerate other large-scale computational applications as well. To that end, the use of reconfigurable-computing technology such as field-programmable gate arrays (FPGAs) is central to our theme of performance and flexibility.

The FPGAs have been widely used to accelerate scientific-computing applications. The ability to reconfigure an FPGA's internal fabric to suit the application's needs means that many applications that perform poorly on traditional architectures are amenable for acceleration on an FPGA. Surveying the Top500 list of supercomputers in the world today [8], we see that 5 of the top 10 supercomputers use accelerators in the form of Intel Xeon Phi coprocessors or Nvidia GPUs. The longer learning curve and turnaround time of FPGA designs have limited their adoption in this respect. Even so, at our center we have had much documented success with our FPGA-centric cluster, Novo-G [9, 10], consisting of nearly 400 Stratix II and Stratix IV FPGAs from Altera. Our work so far on application acceleration in bioinformatics [11–13], image processing [14–16], and financial [17] domains has focused on problems that were challenging to accelerate on a single FPGA but easy to replicate across the cluster. With our recent addition of Stratix V FPGAs, we have extended our Novo-G infrastructure to target communication-intensive problems and applications that would benefit from FPGA acceleration.

In order to efficiently accelerate communication-intensive apps on reconfigurable hardware, we provide a multidimensional backend network that connects to high-speed transceivers on each Stratix V FPGA, enabling high-bandwidth, low-latency communication among the FPGAs. Acceleration kernels on the FPGAs can bypass the host CPU and cluster communication to exchange data and can do so in a distributed manner using the inter-FPGA network. Such multidimensional networks are widely used in conventional supercomputers such as the Titan [18], K-computer [19] and most Blue Gene-based systems [20].

This paper discusses the aforementioned upgrade to Novo-G, a reconfigurable acceleration platform with a 3D torus interconnect that we have dubbed Novo-G# (*novo-jee-sharp*). An initial deployment of Novo-G# consisting of 32 Stratix V FPGAs connected together with 40 Gbps links has been completed, and an upgrade of another 32 nodes is underway. We have designed and developed a 3D torus network stack that enables transparent communication between application kernels running on different FPGAs. We have also developed and tested a 3D fast Fourier transform/inverse fast Fourier transform (FFT/IFFT) kernel on Novo-G# that makes use of inter-FPGA communication. This kernel is the component of MD that interests us the most, because it accounts for a large section of the MD computation time and communication time.

In order to understand the behavior of Novo-G# at scale, we created a simulation model of Novo-G# based on our previous research into modeling the behavior of 3D FFT on the Anton machine [1]. VisualSim [21], a commercial discrete-event simulation and modeling tool, was used because of its simple graphical interface and available library of basic blocks with which to construct models. Our model was constructed using parameters of Novo-G# obtained through hardware design and testing and validated against hardware runs of 3D FFT on eight Novo-G# nodes. Validation of our Anton model against published Anton performance data with less than 7% error at different system sizes showed that the Anton model, and hence the highly similar Novo-G# model, scale well with system size. Based on simulation runs of the validated Novo-G# model, we predict that our system can exhibit 3D FFT performance better than Anton and can outperform conventional systems like the Blue Gene/Q by more than an order of magnitude. Much of this performance gain stems from greater sustained FFT performance because of the acceleration of 3D FFT in hardware, and better inter-FPGA communication performance than the 5D torus used in Blue Gene/Q.

The remainder of this paper is organized as follows. Section 2 gives background information on molecular dynamics, the Anton machine, and Novo-G. Section 3 describes our approach to modeling



DOI: 10.1002/cpe

3D FFT running on the Anton architecture. Section 4 describes the Novo-G# architecture, presents the performance of the inter-FPGA communication links, and details the various components of the reconfigurable system. Section 5 describes the Novo-G# performance model, hardware experiments conducted on the system, and predicted 3D FFT performance for larger system sizes. Finally, Section 6 presents our conclusions and future plans for Novo-G#.

## 2. BACKGROUND

In this section, we briefly review the literature on the molecular-dynamics problem along with an overview of the Anton machine and its architecture. We also describe the development of the Novo-G reconfigurable supercomputer and recent developments in FPGA-based clusters.

### *2.1. Molecular dynamics and the fast Fourier transform kernel*

Molecular dynamics is a computer simulation that models the physical behavior at the molecular or atomic level. MD simulation is frequently used in scientific fields such as biomolecular engineering, neurobiology, and material science. It is well-known for its computation intensity and demanding design requirements for accuracy. Generally, several CPU days to CPU months are needed for a dynamic simulation of DNA or protein molecules, ranging from nanoseconds to microseconds [22]. Therefore, an accuracy-sufficient and time-efficient MD simulation is of importance to research in those fields.

Molecular dynamics software packages, such as NAMD [23] and GROMACS [24], are available for various platforms and scales. In general, the MD algorithm is composed of two parts: computing the interactive forces among particles in the simulation, and deriving their positions and velocities through the integration of those forces. Generally, the performance bottleneck is the force calculation step [25]. The total force for each simulated particle is computed as the sum of bonding forces, which depends on covalent bond structure of particles, and non-bonding forces, which involves the electrostatic and Van der Waals interactions between all pairs of particles in the system.

Calculating the non-bonded forces requires a pairwise computation for every pair of particles in the system. Because the fall-off rate for the Van der Waals force is considerably greater than that of the electrostatic force, a cutoff radius can be applied, thereby simplifying the computation. For long-range forces, such as the electrostatic force, other approximation methods need to be used to reduce the computation time. Among them, the Particle Mesh Ewald (PME) [26] and k-space Gaussian Split Ewald [27] are the most popular. Both methods use a volumetric (3D) FFT to simplify the computation of electrostatic forces. Benchmark results from [20] using GROMACS on various system sizes show that the scalability of long-range force calculation is worse than that of range-limited force calculation, thus making long-range force calculation the most time-intensive part of MD in high-parallelism cases.

In [28], the authors simulate the execution of 3D FFT on a 3D-torus, FPGA-based network for various network sizes using cycle-accurate simulation. They demonstrate that, despite the high degree of communication in the 3D FFT algorithm, the kernel shows strong scaling on small to medium systems and is a good candidate for parallelization on such a system.

A 3D FFT calculation of size $N_x \times N_y \times N_z$ is composed of multiple 1D FFTs and can be divided into three stages: computing $N_y \times N_z$ 1D FFTs of size $N_x$ in the X dimension; then $N_x \times N_z$ 1D FFTs of size $N_y$ in the Y dimension; and finally $N_x \times N_y$ 1D FFTs of size $N_z$ in the Z dimension. The influence of the 3D FFT calculation is negligible when compared with the number of messages required for this parallel implementation, thus limiting the communication scalability of long-range force calculation.

### *2.2. The Anton machine*

Many efforts have been made to accelerate MD simulation on multi-node systems (e.g., Blue Gene/L [29] and MDGRAPE [3]). Among them, the Anton machine, a special-purpose parallel supercomputer, stands out because of processing and communications infrastructure customized for MD. Each Anton node is an application-specific integrated circuit (ASIC) designed specifically for



DOI: 10.1002/cpe

MD. Unlike the MDGRAPE systems that divide MD computation between the host and ASIC processors, on Anton, all MD computation takes place on the ASICs. Anton has also been shown to outperform modern high-performance computing systems because of fast, low-latency, inter-node communication. The data in Table I, collected from various references, compares the performance of Anton with other platforms. Here, a node is the basic element of the system, consisting of a single processing unit (ASIC or CPU) and associated components. The efficient use of hardware in Anton speeds up MD simulation of dihydrofolate reductase (DHFR) by several orders of magnitude.

Figure 1 describes the basic node architecture of the Anton machine. A typical system consists of 512 nodes, with all the nodes connected in an $8 \times 8 \times 8$ torus network. Each node is connected to its six neighbors in six directions with 50.6 Gbps bidirectional links. Each node uses a High-Throughput Interaction Subsystem to calculate range-limited interactions, perform charge spreading, and perform force interpolation. A flexible subsystem is responsible for long-range force calculation, particle updates, and the remainder of the MD pipeline. Every node also includes accumulation memories to sum forces and charges, and a 256-bit bidirectional intra-node ring network that facilitates data movement. Each flexible subsystem in an Anton node contains four processing slices, and eight geometry cores (GCs) that compute the individual 1D FFT stages of the 3D FFT kernel. In a typical MD simulation, Anton spends almost 60% of its time in computing long-range forces [24]. Acceleration of the 3D FFT kernel is therefore the focus of this paper.

Table I. Comparison of MD performance[a] on various platforms.

| MD system | Reference | Nodes | Real time per step (μs) |
|---|---|---|---|
| Anton | [5] | 512 | 19 |
| Desmond on 2.4 GHz AMD | [25] | 256 | 1400 |
| Blue Matter on BG/L | [30] | 8192 | 1700 |
| NAMD on 2.4 GHz AMD | [23] | 128 | 6300 |
| MDGRAPE-3 | [3] | 12 | 26,000 |
| GROMACS on 2.4 GHz AMD | [5] | 1 | 181,000 |

MD, molecular dynamics; DHFR, dihydrofolate reductase.
[a] Measured for DHFR simulation. Table adapted from [4].

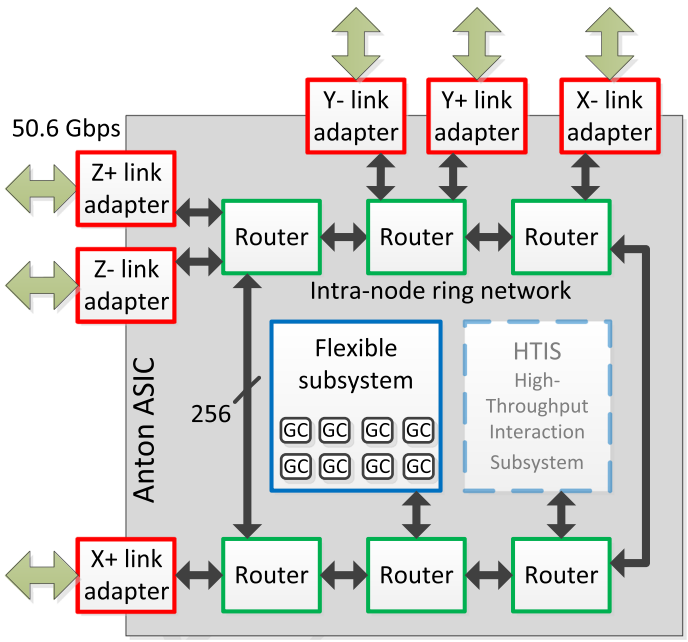


Figure 1. Architecture of Anton machine's ASICs depicting the intra-node ring network. GC: geometry core responsible for 1D FFT computation. ASIC, application-specific integrated circuit; FFT, fast Fourier transform.



DOI: 10.1002/cpe

The D. E. Shaw research lab recently released its first publications on the successor to the Anton machine, Anton2 [6, 31, 32]. A 512-node Anton2 system is currently in operation, with the architecture scaling up to a maximum of 4096 nodes. The new system has demonstrated an order of magnitude performance gain over Anton, because, in part, of the increased number of GCs (64) [6], and aggressive exploitation of fine-grained parallelism in MD. The inter-node and intra-node communication subsystems in Anton2 have been greatly optimized for fine-grained communication and synchronization and provided an effective bidirectional I/O bandwidth of 2.15 Tbps per chip [31]. However, the architectural details provided on Anton2 in the referred publications are insufficient to allow us to construct a fine-grained structural model of its operation. We therefore choose to focus our attention on the more mature Anton machine at this time, until sufficient data on the Anton2 architecture and FFT performance becomes available.

### *2.3. Development of Novo-G*

Novo-G [9, 10] began in 2009 as an effort to create a research cluster using high-density FPGA boards to accelerate scientific applications. The original machine began with a head node and 24 Linux servers, each featuring a quad-FPGA board from Gidel [33], for a total of 96 Altera Stratix III E260 FPGAs. Over subsequent years, the machine has been upgraded annually and now stands at 192 Stratix III FPGAs in 24 servers, 192 Stratix IV E530s housed in 12 servers, and 32 Stratix V GSMD8s in 8 servers. Each server features dual Intel Xeon multicore processors. Connectivity is provided by gigabit Ethernet and DDR/QDR InfiniBand within the system, and a 10 Gbps connection to the Florida LambdaRail.

Novo-G has been the platform of choice for a variety of application-acceleration projects undertaken by the NSF Center for High-Performance Reconfigurable Computing (CHREC). The following are some of the applications developed for Novo-G. Blast-Wrapped Smith–Waterman (bioinformatics) on 128 FPGAs shows a speedup of 50,000 against SSEARCH [11]. Image segmentation (image processing) on four FPGAs shows a speedup of 1106 against an optimized serial baseline [14]. Monte Carlo options pricing (financial computing) on 48 FPGAs shows a speedup of 7134 against an optimized serial baseline [17]. The one common factor among the aforementioned applications is that they are embarrassingly parallel and can therefore scale almost linearly with the available hardware resources. A greater challenge is that of accelerating communication-intensive applications like MD. Traditionally, this communication makes use of centralized networks such as Ethernet or InfiniBand and entails many interactions between the FPGA and the host. Our proposed system would feature a multidimensional network that connects the FPGAs in a 3D torus with bidirectional links, six links per FPGA, at 40 Gbaud per link. For comparison, 4X QDR InfiniBand also provides a signaling rate of 40 Gbaud. A large part of Anton's success with MD comes from the use of a low-latency 3D torus network between processors, and we intend to emulate this facet in Novo-G# with a low-latency 3D torus network connecting the FPGAs together.

In the past year, there has been a growing interest in using reconfigurable hardware in the data center, which has traditionally been slow to integrate new, untested technologies. Microsoft research [34] has integrated Altera FPGAs on custom boards with their pre-existing datacenter infrastructure to accelerate the Bing search engine. Each FPGA accelerator is connected in a $6 \times 8$ torus network to enable the application to be extended beyond a single FPGA. On a medium-scale deployment of 1632 FPGAs, they demonstrated a speedup of two for Bing's document ranking algorithm at an increased power consumption of 10%. In a similar vein, the web services company Baidu [35, 36] has collaborated with Altera to demonstrate FPGA-based acceleration of deep neural networks. This algorithm is a highly accurate method used in key search functions such as image classification and recognition.

## 3. MODELING THE ANTON ARCHITECTURE

In order to develop a model for 3D FFT execution on Anton, we collect published data on the Anton architecture, operation, and decomposition of the 3D FFT algorithm. This section describes the process of modeling the distributed 3D FFT algorithm and the Anton architecture in VisualSim.



DOI: 10.1002/cpe

We validate the developed model against published run times of the distributed 3D FFT on Anton for various system and 3D FFT sizes.

### *3.1. 3D fast Fourier transform application modeling in VisualSim*

As discussed earlier, computation of long-range forces in the MD application is more efficient in Fourier space than real space. Under the PME method, the computation of long-range forces consists of the Fourier transform of the charge-density function, multiplication of the result with the transform of the potential function, and an IFFT of the product. In keeping with published Anton data [37], we only model the FFT and IFFT stages.

The communication behavior of the 3D FFT largely depends on how the data are distributed for each stage of the FFT. For a 16 × 16 × 16 data set processed on a 4 × 4 × 4 system, Figure 2 depicts the communication pattern and FFT stages, using the node at Cartesian coordinates (1, 1, 0) as an example. To succinctly describe the communication pattern for a given FFT and

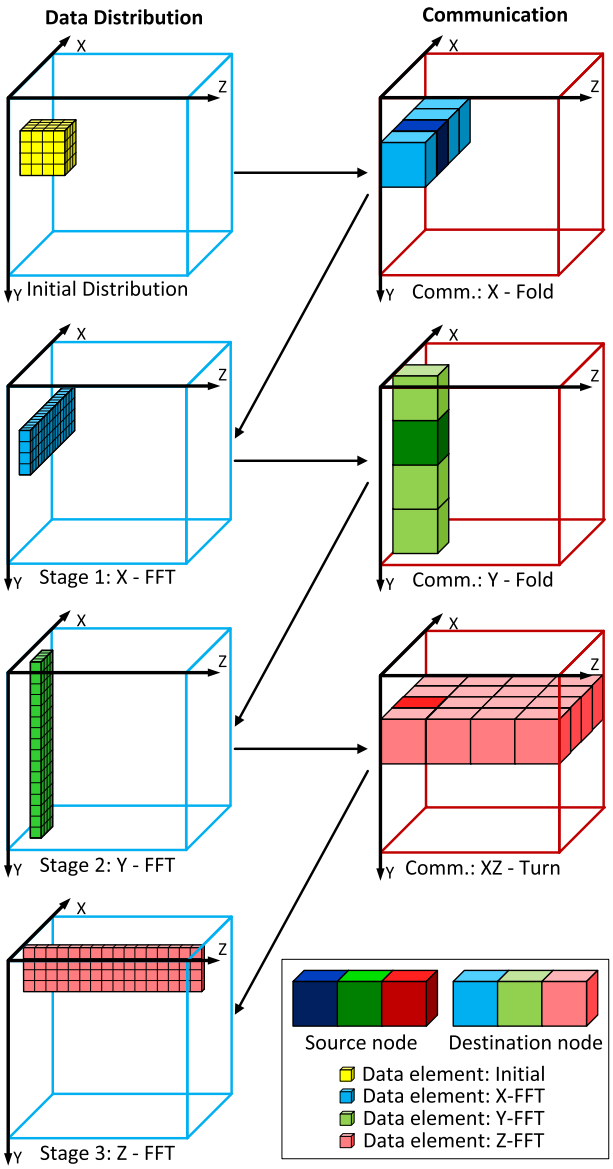


Figure 2. Data cube for a 16 × 16 × 16 point FFT running on a 4 × 4 × 4 system. Data distribution is shown for a single node at Cartesian coordinates (1,1,0). FFT, fast Fourier transform.



DOI: 10.1002/cpe

$(x_4x_3x_2, y_4y_3y_2, z_4z_3z_2).x_1.x_0y_1y_0z_1z_0$ Original Distribution
$(y_1y_0z_0, y_4y_3y_2, z_4z_3z_2).z_1.x_4x_3x_2x_1x_0$ After X - Fold
$(x_1x_0z_0, x_4x_3x_2, z_4z_3z_2).z_1.y_4y_3y_2y_1y_0$ After XY - Turn
$(x_1x_0y_0, x_4x_3x_2, y_4y_3y_2).y_1.z_4z_3z_2z_1z_0$ After XZ - Turn

(a)

```
INIT:Distribute initial FFT data tokens to all nodes
BEGIN:
If input_token corresponds to current FFT stage
    Wait until all data elements arrive at node
    If Verify_mode then
        Compute local FFT
    End If
    For all Data elements in the node
        Generate message for next stage
        Determine destination and data index
        Transmit message delayed by FFT computation time
    End for
End If
```

(b)

Figure 3. (a) Representation of data movement for 32 × 32 × 32 FFT running on 8 × 8 × 8 Anton; (b) Pseudocode for FFT application model in VisualSim. FFT, fast Fourier transform.

system size, we use a modified form of the notation described in [37]. In this notation, a data point $(x_n...x_0, y_n...y_0, z_n...z_0)$, representing its 3D binary address, is mapped to the node with Cartesian coordinates $(Node\_x, Node\_y, Node\_z)$. If index represents the execution unit within the node, and offset represents the data offset within the execution unit, the final mapping for each data point is given by:

$$(x_n...x_0, y_n...y_0, z_n...z_0) \Longleftrightarrow (Node\_x, Node\_y, Node\_z).index.offset$$

As shown in Figure 3(a), this notation can be used to determine the data that need to be transmitted between each node before an FFT computation can begin. The data pattern, and by extension the communication pattern, will change based on the size of the FFT and the system. For the specific case of a 32 × 32 × 32 FFT distributed on an 8 × 8 × 8 Anton system, two 1D FFTs are computed on each node in every stage.

In the VisualSim model, behavior of the 3D FFT is implemented as a script in a virtual machine block, the pseudocode for which is shown in Figure 3(b). Execution of the block is triggered each time an *input_token* is received. We use the *Verify_mode* flag to turn on the code in the script that computes the FFT output. We can thus verify the model by computing the actual FFT outputs for random input data and comparing with the Matlab 3D FFT function. Once verified, the *Verify_mode* switch is turned off to speed up the model.

### *3.2. VisualSim modeling of Anton*

Parameters that we use to model the Anton machine are collected from multiple publications, notably [4, 5, 7, 29, 37, 38], with preference given to recent publications. Table II summarizes these parameters. We built the model in VisualSim as a hierarchical model, with the node and channel models described as independent classes. This method allows us the flexibility of developing the node and channel models independently and modularly. At the top-level, an instance of each class is created for every node in the system being modeled.

The complete VisualSim model of Anton can be found in [1]. The FFT algorithm and routing algorithm are implemented as scripts running inside virtual-machine blocks, while the remainder of the system is modeled using basic blocks from the VisualSim library. To reduce the complexity of the model, we converted the intra-chip ring network, a 256-bit bidirectional bus, to a delay model. To model contention among messages, separate queues are used for each direction of the ring and for the ±Y and ±Z link pairs. Table III summarizes the delay for each source-destination pair.



DOI: 10.1002/cpe

Table II. Modeling parameters for Anton machine.

| Parameter | | Value | References |
|---|---|---|---|
| System frequency | | 485 MHz | [4, 5, 37, 38] |
| Internal bandwidth | | 124.2 Gbps | [4, 5, 29, 38] |
| External bandwidth | | 50.6 Gbps | [4, 5, 7, 29] |
| Synchronization delay | | 42 ns | [38] |
| Package writing delay | | 36 ns | [38] |
| Wire delay | x | 4 ns | [38] |
| | y | 8 ns | [38] |
| | z | 10 ns | [38] |
| Transceiver delay | | 20 ns | [38] |
| FFT calculation time | 1 GC | 137 cycles | [37] |
| | 4 GCs | 75 cycles | [37] |

FFT, fast Fourier transform; GC, geometry core.

Table III. Routing latencies[a] (ns) for Anton machine.

| Destination | Source direction | | | | | | |
|---|---|---|---|---|---|---|---|
| | X+ | X− | Y+ | Y− | Z+ | Z− | Processing slice |
| X+ | — | 31 | 25 | 25 | 19 | 19 | 19 |
| X− | 31 | — | 19 | 19 | 25 | 25 | 25 |
| Y+ | 25 | 19 | — | 13 | 25 | 25 | 31 |
| Y− | 25 | 19 | 13 | — | 19 | 19 | 31 |
| Z+ | 19 | 25 | 25 | 19 | — | 13 | 25 |
| Z− | 19 | 25 | 25 | 19 | 13 | — | 25 |
| Processing slice | 19 | 25 | 31 | 31 | 25 | 25 | — |

[a]Path latencies computed from individual component latencies in [29].

Table IV. Model validation with Anton data.

| System size | FFT size | Parallel strategy | Anton exec. time (μs) | Measured model time (μs) | Error (%) |
|---|---|---|---|---|---|
| $8 \times 8 \times 8$ | $32 \times 32 \times 32$ | 1 FFT: 4 GCs | 3.7 | 3.50 | 5.40 |
| | $32 \times 32 \times 32$ | 1 FFT: 1 GC | 4.0 | 3.75 | 6.25 |
| | $64 \times 64 \times 64$ | 1 FFT: 1 GC | 13.2 | 12.70 | 3.70 |
| $4 \times 4 \times 4$ | $16 \times 16 \times 16$ | 1 FFT: 1 GC | 2.4 | 2.55 | 6.25 |
| | $32 \times 32 \times 32$ | 2 FFTs: 1 GC | 10.5 | 10.00 | 4.80 |

FFT, fast Fourier transform; GC, geometry core.

Anton uses GCs to handle the basic FFT operation. Computing a 32-point 1D FFT on a single GC requires 137 clock cycles. Almost twice the performance (75 cycles) can be achieved by parallelizing every 1D FFT over four GCs using redundant computation [16]. Because 1D FFT is an $\mathcal{O}(N \log N)$ function, we compute the execution time (assuming operation on 1 GC) for a 1D FFT of size *numpoints* as:

$$Cycles_{numpoints} = 137 \times \frac{numpoints \times \log_2 numpoints}{32 \times \log_2 32}$$

### 3.3. Model validation via Anton

We validate the Anton model by comparing simulation times from our model with those reported in [37] by researchers using the Anton machine. The referenced paper provides 3D FFT execution times averaged over 10 runs of a 3D FFT followed by a 3D IFFT. We follow the same procedure in our VisualSim runs.



DOI: 10.1002/cpe

Simulation runs of the Anton model are summarized in Table IV. Because we are leveraging published Anton data, the number of data points available is limited. However, we believe that the FFT size, system size, and parallelization strategy shown here are sufficiently diverse to expose any errors in the model. Given that the FFT data distribution changes with the problem and system size, the communication patterns are different in each case. Special attention is paid to the $64 \times 64 \times 64$ FFT, where the communication pattern is optimized (as described in [37]) to reduce the load on the network. In all cases, we observe that the prediction error of our model is less than 7%.

## 4. THE NOVO-G# SYSTEM

In this section, we describe the recent changes to our existing Novo-G infrastructure that add Stratix V FPGAs with a 3D torus network, enabling direct communication between FPGAs. The low-latency, high-bandwidth network is inspired by the model built in Section 3 and is expected to greatly improve the performance of communication-intensive applications. We also describe the network stack instantiated on each node to allow them to be connected together in a 3D torus. Finally, we describe the design of a distributed 3D FFT/IFFT kernel on eight FPGAs that maps well to the Novo-G# system.

### *4.1. Novo-G# setup*

The Novo-G# system is part of our effort to create an FPGA cluster that can handle communication-intensive applications like MD. In keeping with that theme, the system features ProceV boards, which are PCIe-based accelerator boards from Gidel populated with Stratix V GSMD8 FPGAs from Altera. The GS-series devices are optimized for high-performance, high-bandwidth applications with support up to 36 on-chip transceivers that can operate up to 14.1 Gbaud. Each FPGA is connected to two 8 GB DDR3 SODIMM and two 36-Mbit SRAM memory banks and communicates with the host CPU via PCIe v3.

Our FPGA platform vendor Gidel has provided invaluable assistance by designing a custom daughterboard that allows external access to 24 high-speed transceivers. The transceivers are grouped into six bidirectional links, each link consisting of four parallel channels, enabling the construction of a 3D torus of arbitrary size. Physical connectivity between the boards is provided by a COTS CXP-3QSFP+ split cable that enables each FPGA to be connected in six different directions. Initial deployment of the Novo-G# system was completed in the second half of 2014 with 32 Stratix V boards housed in eight chassis and supporting up to a $2 \times 4 \times 4$ torus. Much as with Novo-G, we plan to expand the Novo-G# system in the future beginning with an upgrade to 64 nodes ($4 \times 4 \times 4$ torus) in 2015.

### *4.2. Novo-G# network architecture*

A part of the hardware resources on each FPGA is used to implement a network stack that services the 3D torus network. The network stack is responsible for accepting data from the application logic, packetizing the data, routing packets across the 3D torus network by the shortest route, and delivering the data to the application logic at its destination. These functions represent a subset of the services provided by the lowest three layers of the Open Systems Interconnection reference model (i.e., physical, data link, and network layers).

Figure 4 depicts our initial implementation of the 3D torus network stack on Novo-G# and the network services associated with each component. The intellectual property (IP) cores used for the transceiver interfaces are primarily supplied by Altera and interface directly with hardware resources attached to each transceiver channel. The remaining blocks are implemented as register transfer level (RTL) code and therefore do comprise a resource overhead for each FPGA. Data generated by the application logic are packetized by the internal receiver block and stored in a first-in, first-out (FIFO) block. Similarly, the external receiver blocks accept packets or streaming data from the transceiver IP. One or more routers are used to route packets from the receiver to the transmitter blocks, which transmit the data further along the network or to the application at the destination node.



DOI: 10.1002/cpe

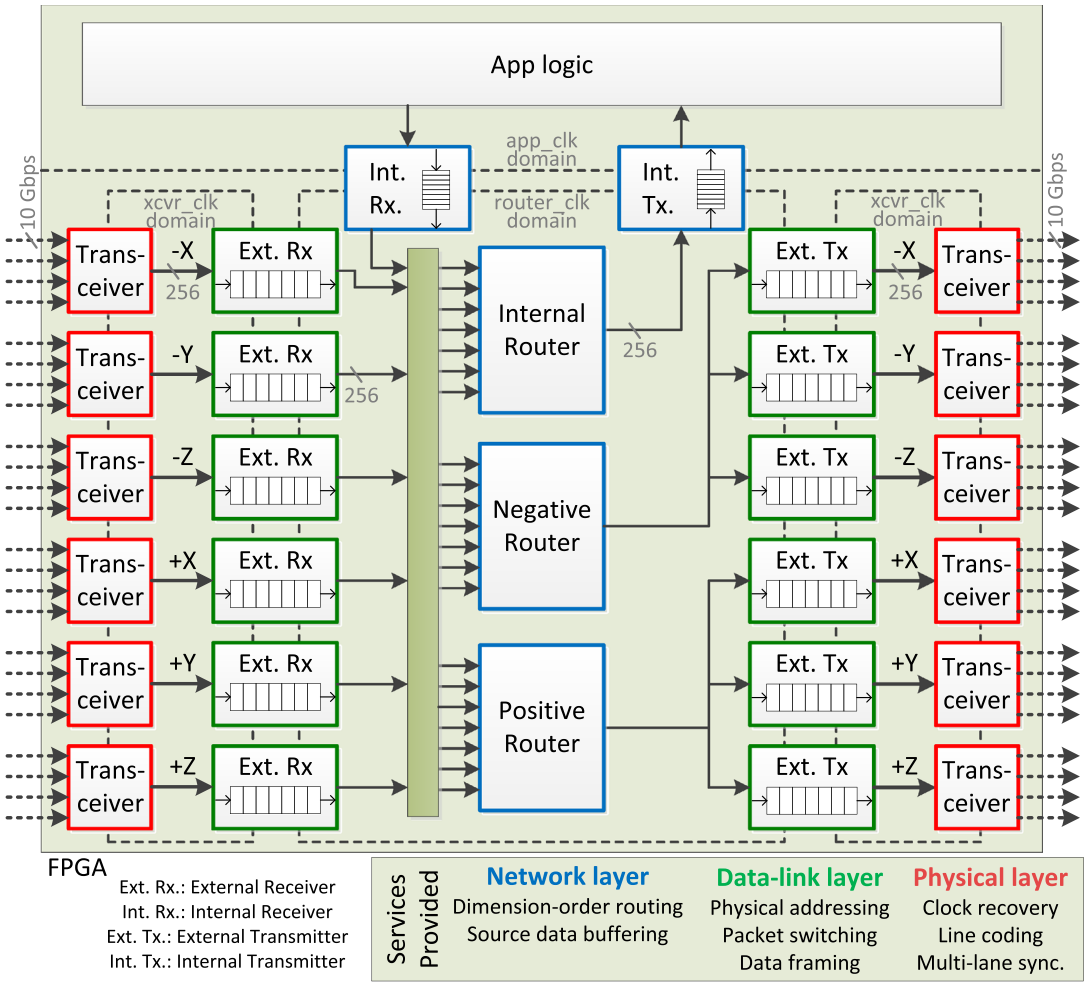


Figure 4. Novo-G# node architecture detailing the 3D torus network stack. Services provided to the user through register transfer level code or third-party intellectual property are also shown.

Traditionally, the router block is the bottleneck in such network architectures and much research has been carried out on optimizing on-chip network architectures. In our previous work on Novo-G# performance prediction [1], we observed that the internal bandwidth provided and latency incurred by the network architecture have the greatest effects on execution time. Distributed architectures like the ring network in [39] scale linearly with the width and number of ports but exhibit high latency because packets may need to traverse half the length of the ring. Anton uses a 256-bit-wide ring network consisting of six routers as shown in Figure 1. Centralized networks such as the crossbar [40, 41] exhibit lower latency, but higher resource usage with a large internal bandwidth.

Our approach uses a centralized single-level router network that can be customized based on hardware resources available and expected utilization of the 3D torus network. Figure 4 shows one such configuration with three routers, each servicing a different subset of output ports. In this manner, more routers can be instantiated (up to the number of output ports) to increase the internal bandwidth available without increasing latency, but at the expense of hardware resources. Further optimization of the router and network design will be explored in future work.

The transceiver blocks in Figure 4 are low-level IP blocks provided by Altera that instantiate various portions of the hardwired transceiver channel. In our aim to make Novo-G# as flexible and reconfigurable as possible, we have evaluated various point-to-point protocols in the hardware. These protocols can be substituted for each other without affecting the rest of the network architecture. Table V summarizes the tested protocols and the features provided by each. The low latency



DOI: 10.1002/cpe

Table V. Comparison of protocol features and latency on Novo-G#.

| Altera PHY IP | Datapath[a] | Data width | Latency (ns)[b] | PHY features |
|---|---|---|---|---|
| Low latency | Standard | 32 | 39 | Phase compensation FIFO, Serializer/deserializer only |
| | 10 G | 64 | 118 | Phase compensation FIFO, Gearbox, Serializer/deserializer only |
| Custom | Standard | 32 | 48 | Low latency plus: 8 b/10 b encoding, Word alignment FSM |
| Interlaken | 10 G | 64 | 118–151 | Low latency plus: 64 b/67 b encoding, Elastic FIFO, multi-lane sync. |

IP, intellectual property; FIFO, first-in, first-out; FSM, finite state machine.
[a]Stratix V transceiver channels can be configured to either hardwired datapath.
[b]Latency measured from input of Tx channel to output to Rx channel in FPGA fabric.

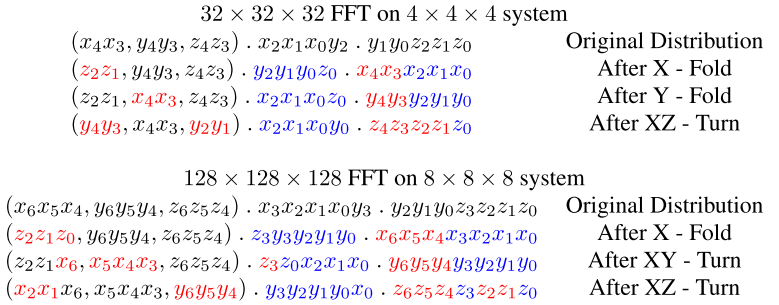


Figure 5. Novo-G# notation. FFT, fast Fourier transform.

PHY IP provides the lowest inter-FPGA latency, but has no word alignment or DC balancing features, making it difficult to use without additional hardware. Conversely, the Interlaken PHY IP does automatic word alignment, 64 b/67 b encoding, scrambling, and multi-lane synchronization, but the inter-FPGA latency is the highest, and varies because of the addition of framing words into the datastream. The Custom PHY provides word alignment through 8 b/10 b encoding and considerably lower inter-FPGA latency than the Interlaken PHY, but the 25% overhead of 8 b/10 b encoding only makes this PHY useful for small packets.

### *4.3. 3D fast Fourier transform design for Stratix V devices*

A number of optimizations to the parallel 3D FFT algorithm shown in Figure 2 can be applied here, considering the flexible Novo-G# architecture. First, while the Anton architecture is optimized for fine-grained communication, Novo-G# does much better with a larger average packet size. Larger packets reduce the inherent overhead associated with adding a header to every packet. In most of the FFT stages, data movement between pairs of nodes can be consolidated as described in [1, 37]. This optimization results in a coarse-grained communication pattern that is more suitable for Novo-G#.

The second optimization is a result of the higher computational density available on the Stratix V devices. Compared with Anton's eight GCs per node, a Novo-G# node can instantiate 16 FFT cores in parallel. The higher computational density allows us to use a smaller system size and reconstruct the communication pattern more efficiently. For example, Figure 5 shows the case of the 32 × 32 × 32 FFT, now distributed on a 4 × 4 × 4 system. On the smaller system, the communication pattern can be modified to use a y-fold (compare with Figure 3(a)) instead of an xy-corner-turn, thus reducing the communication load on the system. Conversely, the smaller system size also leads to longer calculation times because the distribution of 1D FFTs per node is higher.

In our 3D FFT implementation, we use a Nios II soft-core processor [42] connected to the internal transmitter and receiver nodes to service 16 Altera 1D FFT Megacore IPs [43]. The Nios II core



DOI: 10.1002/cpe

is responsible for aggregating FFT inputs coming in from other FPGAs, dispatching FFT data to available IP blocks, and consolidating 1D FFT outputs into packets for transmission to other FPGAs. In hardware, the kernel performs a 3D FFT on the input data present in memory, followed by a 3D IFFT. In the future, additional elements of the MD algorithm such as bonded-force computation and atom exchange may also be implemented in hardware.

## 5. NOVO-G# MODELING AND PERFORMANCE ANALYSIS

In this section, we present results collected from hardware experiments with 3D FFT on eight nodes of Novo-G#. The data collected are used to improve our previously developed Novo-G# model. We can then use the updated model to predict 3D FFT execution times for a variety of system and problem configurations and compare them against existing data for the Anton and BlueGene/Q systems.

### *5.1. Experiments on Novo-G#*

The Novo-G# system currently stands at 32 nodes connected in a $2 \times 4 \times 4$ torus. For this paper, we have limited our hardware experiments to eight FPGAs arranged in a $2 \times 2 \times 2$ torus in order to trace data movement and hardware execution and to extract hardware parameters for use in the performance model. In order to test the hardware execution of 3D FFT on eight nodes, each node is programmed with the 3D torus network architecture and 3D FFT kernel described in Section 4.

Table VI summarizes the hardware experiments conducted, the hardware execution time, and resource utilization of the FFT engine. The memory bits used represents available memory blocks on the device (M20k) and logic blocks used as memory (MLABs). Time and latency measurements taken from various hardware components of the system are used to refine the Novo-G# performance model from [1] and improve its accuracy over the tested configurations. The resultant model is described in the following section, and from the table, we observe less than 2% error on eight nodes. Table VI also provides resource utilization for the Gidel-provided PCIe interface and the 3D torus network stack. We observe that the overhead of using inter-FPGA links on the Stratix V devices has a minimal impact on the resources available for hardware acceleration. The overall FPGA design for a $128 \times 128 \times 128$ 3D FFT kernel on eight nodes of Novo-G# requires less than half the resources available on each FPGA (20% DSP, 14% logic, and 42% memory resources), leaving plenty of resources available for acceleration of other parts of the MD algorithm in the future.

### *5.2. VisualSim modeling of prototype system*

As described in the previous section, the Novo-G# performance model described here is an enhanced version of the model presented in [1]. It retains much of the structure of the validated Anton model, but has been modified to incorporate data and measurements from hardware execution of

Table VI. Hardware experiments on Novo-G#.

| FFT size | Hardware exec. time (μs) | Simulated exec. time (μs) | Relative error (%) | FPGA utilization DSP (%) | Logic (%) | Memory (%) |
|---|---|---|---|---|---|---|
| $16 \times 16 \times 16$ | 3.91 | 3.93 | 0.51 | 7 | 5 | 5 |
| $32 \times 32 \times 32$ | 19.60 | 19.68 | 0.41 | 13 | 5 | 9 |
| $64 \times 64 \times 64$ | 149.10 | 147.60 | −1.01 | 13 | 5 | 18 |
| $128 \times 128 \times 128$ | 1192.00 | 1171.00 | −1.76 | 20 | 5 | 34 |

| | Gidel IP (%) | Physical layer (%) | Router, Rx/Tx (%) | Total FPGA overhead (%) |
|---|---|---|---|---|
| DSP utilization | 0 | 0 | 0 | 0 |
| Logic utilization | 3 | 4 | 2 | 9 |
| Memory utilization | 3 | 1 | 4 | 8 |

FFT, fast Fourier transform; FPGA, field-programmable gate arrays; IP, intellectual property.



DOI: 10.1002/cpe

Table VII. Modeling parameters for Novo-G#.

| Parameter | Value | Notes |
|---|---|---|
| Application layer | | |
| System frequency | 200 MHz | Derived from Altera 1D FFT MegaCore |
| FFT latency | N cycles | data for Stratix V devices; |
| Num_cores | 16 | N = FFT Size |
| Nios core memory latency | 64 cycles | |
| Packet generation latency | 9 cycles | |
| 3D torus network | | |
| Inter-FPGA latency | 134.5 ns | Avg. latency measured with Interlaken PHY |
| Router latency | 3 cycles | From roundtrip latency |
| Ext. Rx/Tx latency | 3 cycles | Altera FIFO latency (frequency optimized) |
| Internal bus width | 256 bits | User-selectable parameter |
| Channel rate | 10 Gbps | Data rate per channel of a link |
| Channel width | 4 | No. of physical channels per link |
| Parameters | | |
| Num_routers | 3 | Router 0: internal port<br>Router 1: negx, negy, negz<br>Router 2: posx, posy, posz |
| System size | — | From $2 \times 2 \times 2$ to $8 \times 8 \times 8$ |
| Packet buffer length | 129 | Input/Output queue length |

3D FFT on the 3D torus. Table VII summarizes the various parameters used by the Novo-G# performance model.

The top-level model shown in Figure 6 represents the entire system through dynamic instantiation blocks. The node model represents an FPGA, and the channel model represents the inter-FPGA links. Individual nodes and channels are instantiated upon starting the simulation, based on the parameters provided to the model. Use of the dynamic instantiation block allows us to simulate systems up to 512 nodes before running out of memory. The top-level model also depicts the various instrumentation and visualization blocks that are used to ensure that the system is performing as expected.

The channel model shown in Figure 7 represents the six input channels to a torus node. In this model, transactions represent individual packets and are queued in each channel for the duration that the packet would require to completely traverse the inter-FPGA link. The primary delay incurred in this model is the average of the inter-FPGA latency observed with the Interlaken PHY described in Table V.

The node model shown in Figure 8 represents the application and 3D torus running on a single node. A script running in the virtual machine in the application layer generates packets based on the pseudo-code in Figure 3(b). The virtual machine block in the Router performs the dimension-order routing of packets towards their destinations. The system of three routers in the hardware is portrayed by the three queues leading up to the router. The numerous delays in this block represent the FFT and application latency, routing latency, and various FIFO block delays. These parameters are summarized in Table VII.

The Novo-G# performance model retains the modularity of the original and can easily be modified to match future optimizations to the system or its components. It can also be used to evaluate design changes before they are implemented on the actual system. In the future, we intend to use this model to evaluate other interconnect strategies, network architectures, and communication protocols. By replacing the existing application layer with another, this model can also easily be used to evaluate the performance and scalability of other applications.



DOI: 10.1002/cpe

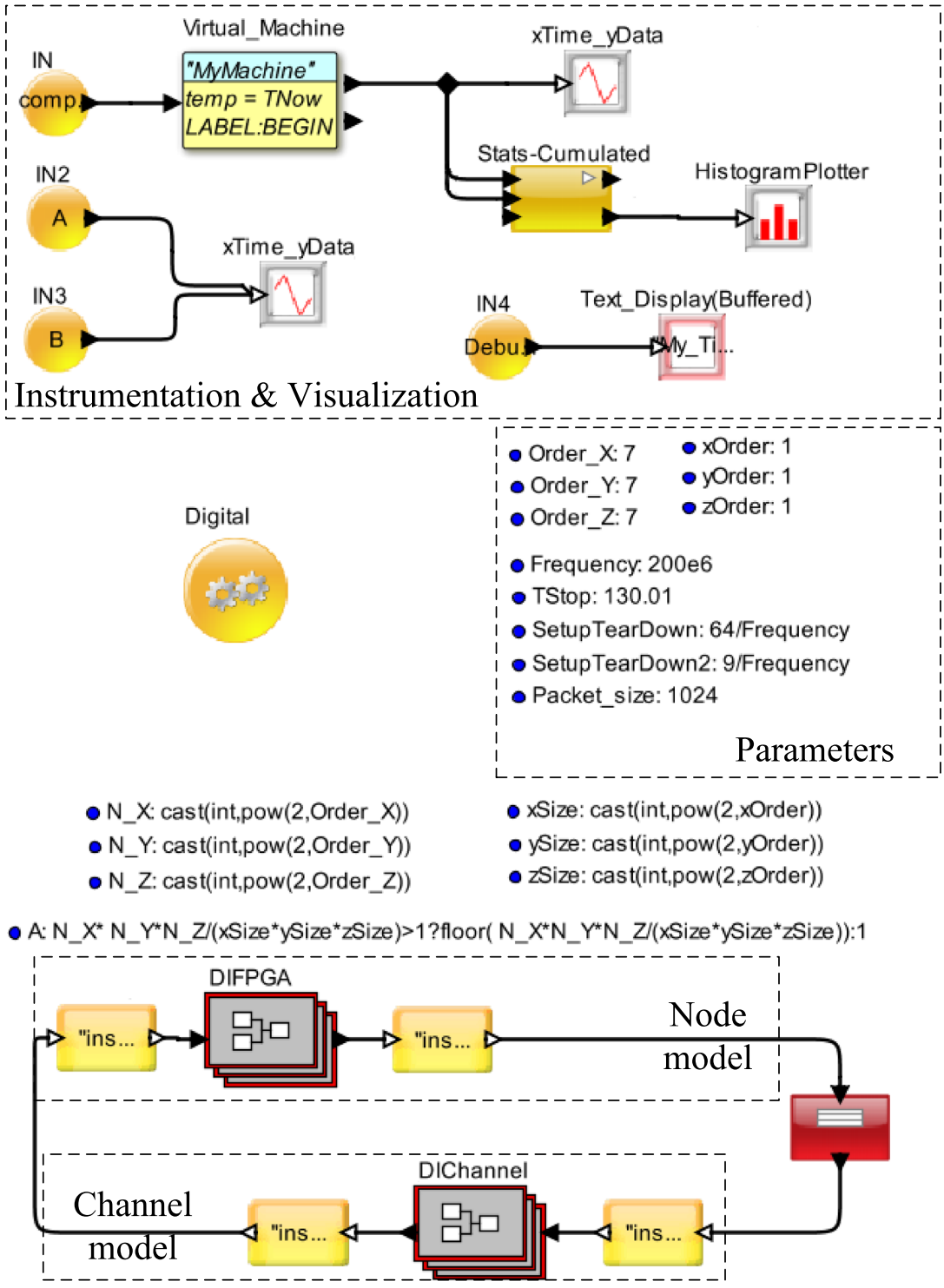


Figure 6. Novo-G# performance model in VisualSim (top level).

### 5.3. *Novo-G# performance predictions*

The Novo-G# performance model presented in the previous section is a structural model that represents the movement of data around the system at a near cycle-accurate level. Each inter-FPGA message is represented in the model as a transaction, which follows the same path and consumes



DOI: 10.1002/cpe

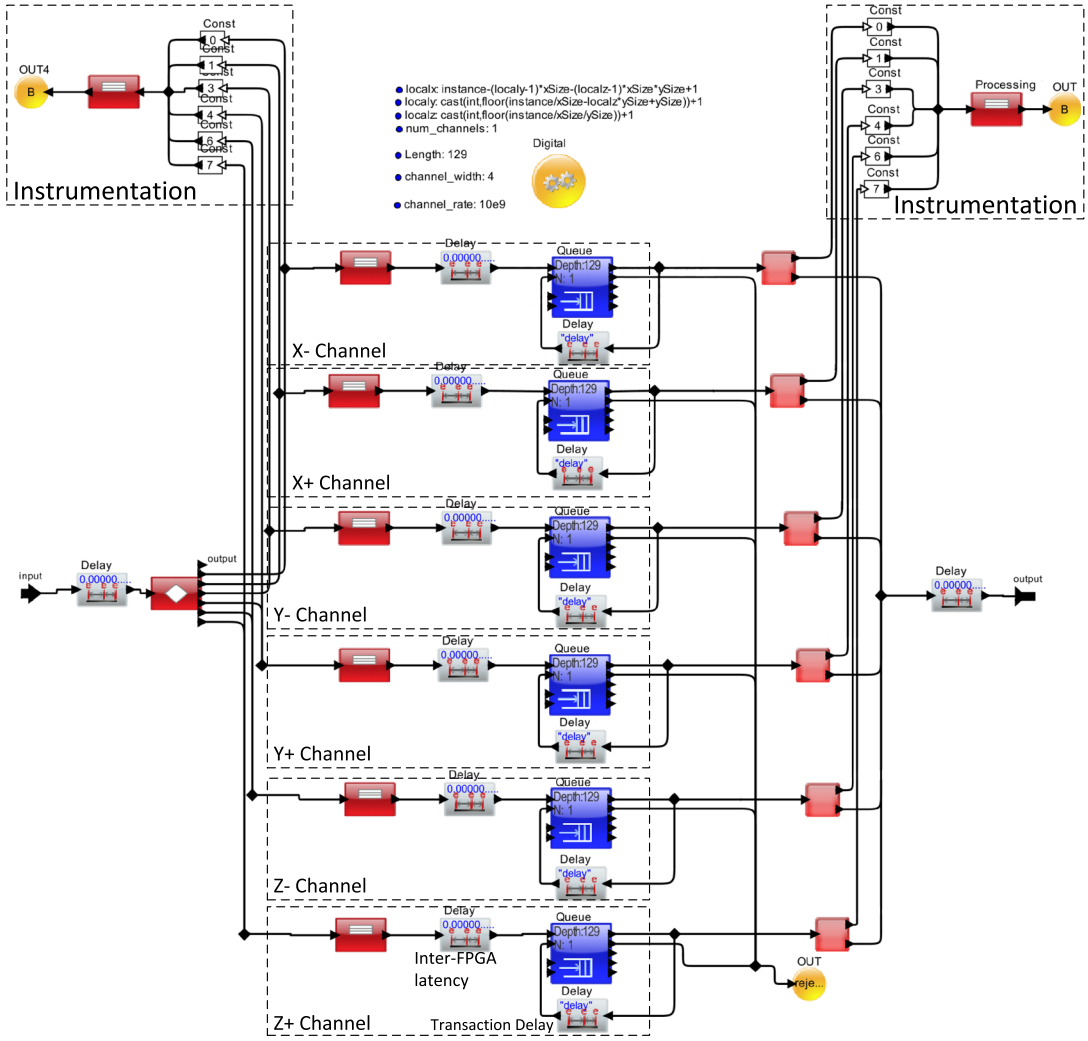


Figure 7. Novo-G# performance model in VisualSim (channel model).

the same network resources that the packet would in hardware. Congestion, deadlocks, and other bottlenecks are therefore modeled precisely. We have shown that, for the 3D FFT/IFFT kernel, the error relative to hardware execution time on a $2 \times 2 \times 2$ torus is less than 2%. We have also shown that the model has a high degree of structural similarity with the Anton model validated at scale with less than 7% error. Thus, we have a high degree of confidence that the Novo-G# performance model can be used to predict the 3D FFT/IFFT performance at scale.

Table VIII shows the simulation results for Novo-G# up to 512 nodes. In each case, a communication pattern appropriate for the given FFT and system size is chosen. The $2 \times 4 \times 2$, $2 \times 4 \times 4$, $4 \times 4 \times 8$, and $4 \times 8 \times 8$ torus configurations are selected for 16, 32, 128, and 256 nodes, respectively, as these configurations show the lowest execution time for the majority of FFT sizes. The case of $16 \times 16 \times 16$ FFT on an $8 \times 8 \times 8$ system is omitted because the FFT distribution per node is less than one. The chart in Figure 9(a) shows that execution times of the $16 \times 16 \times 16$ and $32 \times 32 \times 32$ FFTs reach a minimum at 16 and 128 nodes, respectively. At larger system sizes, the increased communication load and reduced computation load causes the overall execution time to increase.

Figure 9(b) charts the ratio of ideal FFT computation time to the predicted time for Novo-G# for various FFT and system configurations. In other words, this graph represents the parallel efficiency of 3D FFT on Novo-G# . We observe that the parallel efficiency reduces considerably beyond a



DOI: 10.1002/cpe

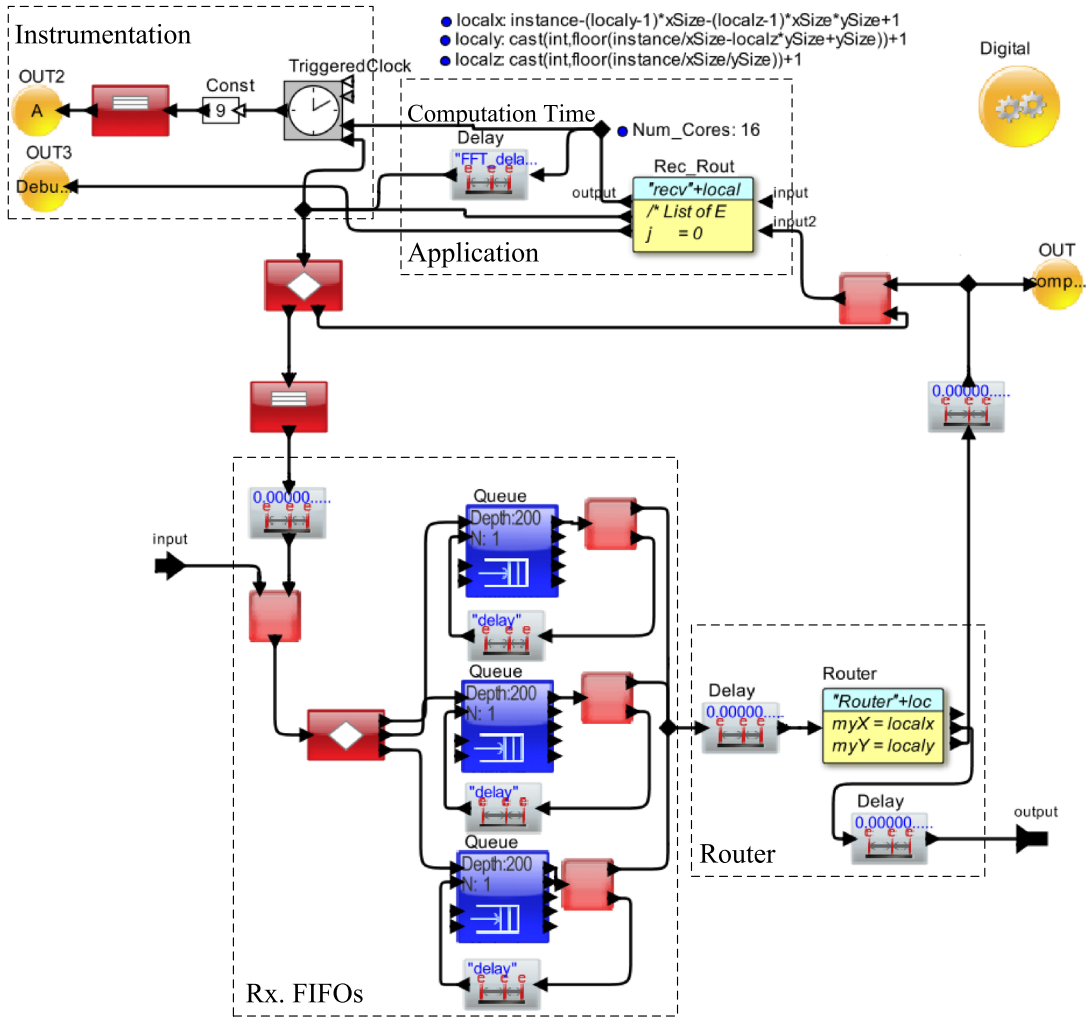


Figure 8. Novo-G# performance model in VisualSim (node model).

Table VIII. Predicted 3D FFT/IFFT kernel execution times (μs).

| | System size (nodes) | | | | | | |
|---|---|---|---|---|---|---|---|
| FFT size | 2 × 2 × 2 (8) | 2 × 4 × 2 (16) | 2 × 4 × 4 (32) | 4 × 4 × 4 (64) | 4 × 4 × 8 (128) | 4 × 8 × 8 (256) | 8 × 8 × 8 (512) |
| 16 × 16 × 16 | 3.93 | 3.66 | 3.80 | 4.51 | 5.20 | 5.94 | — |
| 32 × 32 × 32 | 19.68 | 14.57 | 9.89 | 7.46 | 6.71 | 6.93 | 8.25 |
| 64 × 64 × 64 | 147.60 | 107.60 | 61.75 | 39.25 | 25.52 | 16.11 | 12.64 |
| 128 × 128 × 128 | 1171.00 | 844.40 | 482.40 | 298.10 | 173.90 | 108.60 | 65.11 |

FFT, fast Fourier transform; IFFT, inverse fast Fourier transform.

2 × 2 × 2 system because of the all-to-all communication pattern involved in the 3D FFT algorithm. There is a diminishing return on increasing the hardware resources used to obtain the smallest possible execution time. Optimizations to the Novo-G# network to take advantage of the all-to-all communication pattern and better support for collective operations will greatly improve the performance of 3D FFT, and consequently the performance of MD as well.

As discussed earlier, many high-end supercomputers are designed to run computational science apps such as MD, and also make use of multidimensional interconnects. Of particular interest are the Blue Gene/Q (BG/Q) systems that currently comprise four of the top ten supercomputers on the



DOI: 10.1002/cpe

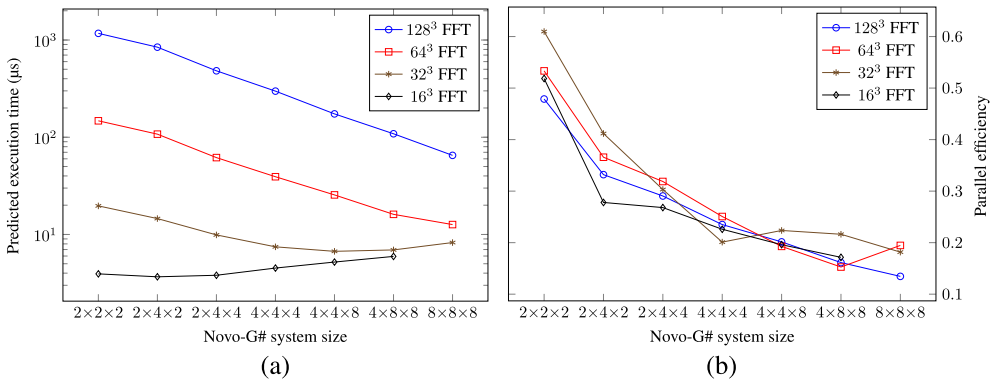


Figure 9. (a) Predicted 3D FFT/IFFT run times on Novo-G#. (b) Parallel efficiency of 3D FFT on Novo-G#. FFT, fast Fourier transform; IFFT, inverse fast Fourier transform.

Table IX. The 3D FFT/IFFT performance on BG/Q (µs) and Novo-G# speedup against BG/Q.

| | 64 nodes | | 128 nodes | | 256 nodes | | 512 nodes | |
|---|---|---|---|---|---|---|---|---|
| FFT size | BG/Q (µs) | Speedup | BG/Q (µs) | Speedup | BG/Q (µs) | Speedup | BG/Q (µs) | Speedup |
| 32 × 32 × 32 | 142 | 19.03 | 127 | 18.94 | 110 | 15.86 | 93 | 11.26 |
| 64 × 64 × 64 | 507 | 12.92 | 459 | 17.99 | 268 | 16.64 | 229 | 18.12 |
| 128 × 128 × 128 | 1826 | 6.13 | 1426 | 8.20 | 944 | 8.69 | 677 | 10.40 |

FFT, fast Fourier transform; IFFT, inverse fast Fourier transform.

Top500 list [8]. The BG/Q system is designed by IBM [44] and uses an IBM PowerPC A2 processor with a peak performance of 204.8 GFLOPS and a 5D torus interconnect between nodes, making it particularly suited for MD.

The work in [20] describes a 3D FFT benchmark developed in Charm++ and executed on 1024 nodes of a BG/Q system and the optimizations made to communication primitives used by the algorithm. The optimized execution times and corresponding Novo-G# speedup are presented in Table IX. We observe that Novo-G# is predicted to attain an order of magnitude speedup over the BG/Q system up to 512 nodes, which is the limit of our performance model. From their data, we can see that the higher dimensionality network in the BG/Q system helps with scaling smaller FFTs such as 32 × 32 × 32. The optimizations used in [20] involve the use of many-to-many communication primitives over point-to-point primitives, and we have already seen from Figure 9(b) that Novo-G# could greatly benefit from such optimization as well. The greater bandwidth provided by Novo-G#'s 3D torus and the sustained computational performance (estimated at 100+ GFLOPS per FPGA) contribute to the speedup seen in Table IX. Comparing with Anton data in Table IV, we see that Novo-G# can achieve up to twice the performance of an ASIC system designed for MD.

## 6. CONCLUSIONS

In this paper, we described Novo-G#, a reconfigurable cluster with a multidimensional interconnect. The first iteration of Novo-G# was deployed in the fall of 2014 with 32 nodes, and an upgrade of another 32 nodes is underway. Our goal in building Novo-G# is to build upon our success in accelerating RC-amenable apps on Novo-G, and focus on the hardware acceleration of communication-intensive applications like MD. We described the Novo-G# network architecture and our flexible three-layer network stack that makes inter-FPGA communication possible. At the heart of the architecture is a fast, customizable routing infrastructure consisting of multiple routers. The network architecture has minimal impact on resources available for hardware acceleration, with the entire network consuming only about 9% of FPGA resources.



DOI: 10.1002/cpe

The 3D or volumetric FFT is the dominant kernel of the molecular-dynamics application. MD being a frequently used, large-scale, data-movement problem, there have been many attempts to accelerate it on conventional clusters such as Blue Gene/Q and on unconventional and specialized systems such as Anton. We used novel architectural and algorithmic techniques to design a parallel 3D FFT kernel in hardware that can take advantage of the high-bandwidth, low-latency, multidimensional connectivity provided by Novo-G#. This kernel was tested on eight nodes of the system connected in a $2 \times 2 \times 2$ network configuration.

In order to evaluate the performance of Novo-G#, we presented a discrete-event simulation model for Novo-G# that can predict 3D FFT performance on up to 512 nodes of the system. The model incorporates data from hardware experiments on eight nodes and shows less than 2% error against them. The Novo-G# model also shares a high degree of structural similarity with our previously developed Anton model, which was validated at scale against published Anton data within 7% error. We can therefore use our Novo-G# performance model to predict 3D FFT execution times with a high degree of confidence. Simulation studies showed 3D FFT execution on Novo-G# to be twice as fast as Anton. The same performance is predicted to be up to 20 times as fast when compared with conventional supercomputers such as Blue Gene/Q.

A major factor that contributes to this performance is the low-latency, high-bandwidth network for Novo-G# that is inspired by Anton and the computational density provided by the Stratix V devices. Combined with tight integration with the network interfaces and a flexible design, these factors result in a system that can successfully be used to accelerate MD and other computational science applications. Moving forward, the developed models will be used to design and evaluate better intra-node routing and protocols for use in Novo-G# with the goal of developing a reconfigurable and sustainable cluster for acceleration of a broad variety of communication-intensive applications.

The design of our Novo-G# system was primarily guided by the requirements of MD acceleration, but there are many communication-intensive scientific problems that can benefit from such an architecture. Moving forward, we intend to investigate applications from other scientific domains while continuing development and optimization of MD for Novo-G#. The VisualSim model presented here can easily be adapted to evaluate the amenability and predict the performance of such kernels and applications. Finally, we intend to model and design better intra-FPGA network architectures and communication protocols, continue to expand Novo-G# to larger system sizes, and improve its capability as an acceleration platform.

## ACKNOWLEDGEMENTS

This work was supported in part by the I/UCRC Program of the National Science Foundation under grant nos. EEC-0642422 and IIP-1161022. We gratefully acknowledge the tools and devices provided by Mirabilis Design and Altera, respectively. We especially acknowledge the extended hardware support provided by Gidel.

## REFERENCES

1. Lawande A, Yang H, George A, Lam H. Simulative analysis of a multidimensional torus-based reconfigurable cluster for molecular dynamics. In *Proceedings of the 43rd International Conference on Parallel Processing Workshops - ICCPW '14*. IEEE Press: Piscataway, NJ, USA, 2014; 387–394. DOI: 10.1109/ICPPW.2014.58.
2. Bourlioux A, Gander MJ, Sabidussi G (eds.) *Modern Methods in Scientific Computing and Applications*. Springer: Netherlands, 2002. DOI: 10.1007/978-94-010-0510-4.
3. Narumi T, Taiji M, Ikei M, Ohno Y, Okimoto N, Koishi T, Suenaga A, Futatsugi N, Yanai R, Himeno R, Fujikawa S. Gordon Bell finalists II—a 55 TFLOPS simulation of amyloid-forming peptides from yeast prion Sup35 with the special-purpose computer system MDGRAPE-3. In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing - SC '06*. ACM Press: New York, New York, USA, 2006; (Articlle no. 49). DOI: 10.1145/1188455.1188506.
4. Shaw DE, Chao JC, Eastwood MP, Gagliardo J, Grossman JP, Ho CR, Ierardi DJ, Kolossváry István, Klepeis JL, Layman T, McLeavey C, Deneroff MM, Moraes MA, Mueller R, Priest EC, Shan Y, Spengler J, Theobald M, Towles B, Wang SC, Dror RO, Kuskin JS, Larson RH, Salmon JK, Young C, Batson B, Bowers KJ. Anton, a special-purpose machine for molecular dynamics simulation. In *Proceedings of the 34th Annual International Symposium on Computer Architecture - ISCA '07*. ACM Press: New York, New York, USA, 2007; 1–12. DOI: 10.1145/1250662.1250664.

5. Shaw DE, Chao JC, Eastwood MP, Gagliardo J, Grossman JP, Ho CR, Lerardi DJ, Kolossváry I, Klepeis JL, Layman T, McLeavey C, Deneroff MM, Moraes MA, Mueller R, Priest EC, Shan Y, Spengler J, Theobald M, Towles B, Wang SC, Dror RO, Kuskin JS, Larson RH, Salmon JK, Young C, Batson B, Bowers KJ. Anton, a special-purpose machine for molecular dynamics simulation. *Communications of the ACM* 2008; **51**(7):91–97. DOI: 10.1145/1364782.1364802.
6. Shaw DE, Grossman JP, Bank JA, Batson B, Butts JA, Chao JC, Deneroff MM, Dror RO, Even A, Fenton CH, Forte A, Gagliardo J, Gill G, Greskamp B, Ho CR, Ierardi DJ, Iserovich L, Kuskin JS, Larson RH, Layman T, Lee LS, Lerer AK, Li C, Killebrew D, Mackenzie KM, Mok SYH, Moraes MA, Mueller R, Nociolo LJ, Peticolas JL, Quan T, Ramot D, Salmon JK, Scarpazza DP, Ben Schafer U, Siddique N, Snyder CW, Spengler J, Tang PTP, Theobald M, Toma H, Towles B, Vitale B, Wang SC, Young C. 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*, SC '14. IEEE Press: Piscataway, NJ, USA, 2014; 41–53. DOI:10.1109/SC.2014.9.
7. Shaw DE, Bowers KJ, Chow E, Eastwood MP, Ierardi DJ, Klepeis JL, Kuskin JS, Larson RH, Lindorff-Larsen K, Maragakis P, Moraes MA, Dror RO, Piana S, Shan Y, Towles B, Salmon JK, Grossman JP, Mackenzie KM, Bank JA, Young C, Deneroff MM, Batson B. Millisecond-scale molecular dynamics simulations on Anton. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis - SC '09*. ACM Press: New York, New York, USA, 2009; 1–11. DOI: 10.1145/1654059.1654126.
8. November 2014 | Top 500 Supercomputer Sites. (Available from: http://www.top500.org/lists/2014/11/) [Accessed on 25 February 2015].
9. George AD, Lam H, Lawande A, Pascoe C, Stitt G. Novo-g: a view at the HPC crossroads for scientific computing. *ERSA*, 2010; 21–30.
10. George A, Lam H, Stitt G. Novo-G: at the forefront of scalable reconfigurable supercomputing. *Computing in Science & Engineering* 2011; **13**(1):82–86.
11. Lam BC, Pascoe C, Schaecher S, Lam H, George AD. BSW: FPGA-accelerated BLAST-wrapped Smith–Waterman aligner. In *2013 International Conference on Reconfigurable Computing and FPGAs (ReconFig)*. IEEE Press: Piscataway, NJ, USA, 2013; 1–7. DOI: 10.1109/ReConFig.2013.6732273.
12. Pascoe C, Lawande A, Lam H, George A, Sun Y, Farmerie W, Herbordt M. Reconfigurable supercomputing with scalable systolic arrays and in-stream control for wavefront genomics processing. *Proceedings of Symposium on Application Accelerators in High-Performance Computing*, Cancun, Mexico, 2010; 13–15.
13. Pascoe C, Box D, Lam H, George A. FPGA-accelerated isotope pattern calculator for use in simulated mass spectrometry peptide and protein chemistry. In *2012 Symposium on Application Accelerators in High Performance Computing*. IEEE Press: Piscataway, NJ, USA, 2012; 111–120. DOI: 10.1109/SAAHPC.2012.9.
14. Craciun S, Wang G, George AD, Lam H, Principe JC. A scalable RC architecture for mean-shift clustering. In *2013 IEEE 24th International conference on Application-Specific Systems, Architectures and Processors*. IEEE Press: Piscataway, NJ, USA, 2013; 370–374. DOI: 10.1109/ASAP.2013.6567603.
15. Zicari P, Lam H, George A. Reconfigurable computing architecture for accurate disparity map calculation in real-time stereo vision. *International Conference on Image Processing, Computer Vision, and Pattern Recognition*, Las Vegas, Nevada, USA, 2013.
16. Craciun S, George AD, Lam H, Principe JC. A parallel hardware architecture for information-theoretic adaptive filtering. In *2010 Fourth International Workshop on High-Performance Reconfigurable Computing Technology and Applications (HPRCTA)*. IEEE Press: Piscataway, NJ, USA, 2010; 1–10. DOI: 10.1109/HPRCTA.2010.5670798.
17. Sridharan R, Cooke G, Hill K, Lam H, George A. FPGA-based reconfigurable computing for pricing multi-asset barrier options. In *2012 Symposium on Application Accelerators in High Performance Computing*. IEEE Press: Piscataway, NJ, USA, 2012; 34–43. DOI: 10.1109/SAAHPC.2012.21.
18. Vishnu A, ten Bruggencate M, Olson R. Evaluating the potential of Cray Gemini interconnect for PGAS communication runtime systems. In *2011 IEEE 19th Annual Symposium on High Performance Interconnects (HOTI)*. IEEE Press: Piscataway, NJ, USA, 2011; 70–77.
19. Ajima Y, Takagi Y, Inoue T, Hiramoto S, Shimizu T. The Tofu interconnect. In *2011 IEEE 19th Annual Symposium on High Performance Interconnects (HOTI)*. IEEE Press: Piscataway, NJ, USA, 2011; 87–94.
20. Kumar S, Sun Y, Kale L. Acceleration of an asynchronous message driven programming paradigm on IBM Blue Gene/Q. *2013 IEEE 27th International Symposium on Parallel Distributed Processing (IPDPS)*, Boston, Massachusetts, USA, 2013; 689–699. DOI: 10.1109/IPDPS.2013.83.
21. *VisualSim | Mirabilis Design*. (Available from: http://www.mirabilisdesign.com/new/visualsim/) [Accessed on 25 February 2015].
22. Alder BJ, Wainwright TE. Studies in molecular dynamics. I. General method. *The Journal of Chemical Physics* 1959; **31**(2):459–466. American Institute of Physics: New York, NY, USA.
23. Phillips JC, Braun R, Wang W, Gumbart J, Tajkhorshid E, Villa E, Chipot C, Skeel RD, Kalé L, Schulten K. Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry* 2005; **26**(16):1781–802.
24. Berendsen H, van der Spoel D, van Drunen R. GROMACS: a message-passing parallel molecular dynamics implementation. *Computer Physics Communications* 1995; **91**(1–3):43–56.
25. Bowers K, Chow E, Xu H, Dror R, Eastwood M, Gregersen B, Klepeis J, Kolossvary I, Moraes M, Sacerdoti F, Salmon J, Shan Y, Shaw D. Scalable algorithms for molecular dynamics simulations on commodity clusters. In *ACM/IEEE SC 2006 Conference (SC'06)*. IEEE Press: Piscataway, NJ, USA, 2006; 43–43. DOI: 10.1109/SC.2006.54.

26. Darden T, York D, Pedersen L. Particle mesh Ewald: an N.log(N) method for Ewald sums in large systems. *The Journal of Chemical Physics* 1993; **98**(12):10089–10092. American Institute of Physics: New York, NY, USA.
27. Shan Y, Klepeis JL, Eastwood MP, Dror RO, Shaw DE. Gaussian split Ewald: a fast Ewald mesh method for molecular simulation. *The Journal of chemical physics* 2005; **122**(5):54101-1–54101-13. American Institute of Physics: New York, NY, USA Article no. 54101.
28. Sheng J, Humphries B, Zhang H, Herbordt MC. Design of 3D FFTs with FPGA clusters. *2014 IEEE High Performance Extreme Computing Conference (HPEC'14)*, Waltham, MA, USA, 2014.
29. Kuskin JS, Young C, Grossman JP, Batson B, Deneroff MM, Dror RO, Shaw DE. Incorporating flexibility in Anton, a specialized machine for molecular dynamics simulation. In *2008 IEEE 14TH International Symposium on High Performance Computer Architecture*. IEEE Press: Piscataway, NJ, USA, 2008; 343–354. DOI: 10.1109/HPCA.2008.4658651.
30. Fitch BG, Rayshubskiy A, Eleftheriou M, Ward TJC, Giampapa M, Zhestkov Y, Pitman MC, Suits F, Grossfield A, Pitera J, Swope W, Zhou R, Feller ScR. Blue matter: strong scaling of molecular dynamics on Blue Gene/L. In *Computational Science - ICCS 2006*. Springer: Berlin Heidelberg, 2006; 846–854. DOI: 10.1007/11758525_113.
31. Towles B, Grossman J, Greskamp B, Shaw D. Unifying on-chip and inter-node switching within the Anton 2 network. *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, Minneapolis, MN, USA, 2014; 1–12. DOI: 10.1109/ISCA.2014.6853238.
32. Grossman JP, Kuskin JS, Bank JA, Theobald M, Dror RO, Ierardi DJ, Larson RH, Schafer UB, Towles B, Young C, Shaw DE. Hardware support for fine-grained event-driven computation in Anton 2. In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '13. ACM: New York, NY, USA, 2013; 549–560. DOI: 10.1145/2451116.2451175.
33. Gidel Products. (Available from: http://www.gidel.com/Products.htm) [Accessed on 25 February 2015].
34. Putnam A, Caulfield AM, Chung ES, Chiou D, Constantinides K, Demme J, Esmaeilzadeh H, Fowers J, Prashanth G, Jan G, Michael G, Hauck HS, Heil S, Hormati A, Kim J-Y, Lanka S, Larus J, Peterson E, Pope S, Smith A, Thong J, Yi P, Burger XD. A reconfigurable fabric for accelerating large-scale datacenter services. In *Proceedings of the 41st Annual International Symposium on Computer Architecuture*, ISCA '14. IEEE Press: Piscataway, NJ, USA, 2014; 13–24. (Available from: http://dl.acm.org/citation.cfm?id=2665671.2665678).
35. Altera and Baidu Collaborate on FPGA-based Acceleration for Cloud Data Centers | Press Release, 2014. (Available from: http://newsroom.altera.com/press-releases/altera-baidu-fpga-cloud-data-centers.htm) [Accessed on 25 February 2015].
36. Ouyang J, Lin S, Qi W, Wang Y, Yu B, Jiang S. SDA: software-defined accelerator for large-scale DNN systems. *Hot Chips: A Symposium on High Performance Chips, 2014*, Stanford, CA, USA, 2014. (Available from: http://www.hotchips.org/wp-content/uploads/hc_archives/hc26/HC26-12-day2-epub/HC26.12-5-FPGAs-epub/HC26.12.545-Soft-Def-Acc-Ouyang-baidu-v3--baidu-v4.pdf).
37. Young C, Bank JA, Dror RO, Grossman JP, Salmon JK, Shaw DE. 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, Storage and Analysis - SC '09*. ACM Press: New York, New York, USA, 2009; 1–11. DOI: 10.1145/1654059.1654083.
38. Dror RO, Grossman J, Mackenzie KM, Towles B, Chow E, Salmon JK, Young C, Bank Ja, Batson B, Deneroff MM, Kuskin JS, Larson RH, Moraes Ma, Shaw DE. Exploiting 162-nanosecond end-to-end communication latency on Anton. In *2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press: Piscataway, NJ, USA, 2010; 1–12. DOI: 10.1109/SC.2010.23.
39. Kim J, Kim H. Router microarchitecture and scalability of ring topology in on-chip networks. In *Proceedings of the 2nd International Workshop on Network on Chip Architectures*, NoCArc '09. ACM: New York, NY, USA, 2009; 5–10. DOI: 10.1145/1645213.1645217.
40. Kim D, Lee K, Lee SJ, Yoo HJ. A reconfigurable crossbar switch with adaptive bandwidth control for networks-on-chip. *IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005*, Vol. 3, Kobe, Japan, 2005; 2369–2372. DOI: 10.1109/ISCAS.2005.1465101.
41. Dai Z, Zhu J. Saturating the transceiver bandwidth: switch fabric design on FPGAS. In *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, FPGA '12. ACM: New York, NY, USA, 2012; 67–76. DOI: 10.1145/2145694.2145706.
42. Embedded Processor | Altera. (Available from: http://www.altera.com/devices/processor/nios2/ni2-index.html), [Accessed on 25 February 2015].
43. FFT Megacore Function | User Guide. (Available from: http://www.altera.com/literature/ug/ug_fft.pdf), [Accessed on 25 February 2015].
44. IBM System Blue Gene Solution. (Available from: http://www-03.ibm.com/systems/technicalcomputing/solutions/bluegene/) [Accessed on 25 February 2015].