Enhancing HPC Education and Workflows with Novel Computing Architectures

Jeffrey Young
Georgia Institute of Technology
Atlanta, Georgia
jyoung9@gatech.edu

Aaron Jezghani
Georgia Institute of Technology
Atlanta, Georgia
ajezghani3@gatech.edu

Jeffrey Valdez
Georgia Institute of Technology
Atlanta, Georgia
valdez@cc.gatech.edu

Sam Jijina
Georgia Institute of Technology
Atlanta, Georgia
sam.jijina@gatech.edu

Xueyang Liu
Georgia Institute of Technology
Atlanta, Georgia
xliu791@gatech.edu

Michael D. Weiner
Georgia Institute of Technology
Atlanta, Georgia
mweiner3@gatech.edu

Will Powell
Georgia Institute of Technology
Atlanta, Georgia
will.powell@cc.gatech.edu

Semir Sarajlic
Georgia Institute of Technology
Atlanta, Georgia
semir.sarajlic@oit.gatech.edu

ABSTRACT

Recent HPC education efforts have focused on maximizing the usage of traditional- and cloud-based computing infrastructures that primarily support CPU or GPU hardware. However, recent innovations in CPU architectures from Arm and RISC-V and the acquisition of Field-Programmable Gate Array (FPGA) companies by vendors like Intel and AMD mean that traditional HPC clusters are rapidly becoming more heterogeneous.

This work investigates one such example deployed at Georgia Tech - a joint workflow for processor design and reconfigurable computing courses supported by both the HPC-focused Partnership for an Advanced Computing Environment (PACE) and GT's novel architecture center, CRNCH. This collaborative workflow of HPC nodes and 40 remotely accessible Pynq devices supported over 100 students in Spring 2022, and its deployment provides key lessons on sticking points and opportunities for combined HPC and novel architecture workflows.

KEYWORDS

HPC education, novel architecture workflows, job scheduling, Field-Programmable Gate Arrays, Jupyter notebooks

1 INTRODUCTION

Many HPC education efforts have focused on extending student experiences from the traditional CPU-based systems found in their phones and laptops to novel accelerators like GPUs for applications like graphics programming and machine learning. At the same time, there are more instances of novel hardware that do not easily fit into our current understanding of what an HPC cluster is including quantum computers, neuromorphic processors, and reconfigurable computing platforms like Field Programmable Gate Arrays (FPGAs).

These new types of platforms can vary from accelerators (like GPUs) to self-hosted devices like NVIDIA's Jetson platform or Xilinx's PYNQ Z-2 [29] board. In the case of the Xilinx PYNQ device, an $150 board provides a small FPGA fabric as well as a two-core Arm 32-bit CPU that can run a full Ubuntu operating system. Typically these boards are used by students in a hands-on fashion where

Figure 1: PYNQ Cluster as part of Georgia Tech's CRNCH Rogues Gallery Testbed

https://doi.org/10.22369/issn.2153-4136/x/x/x

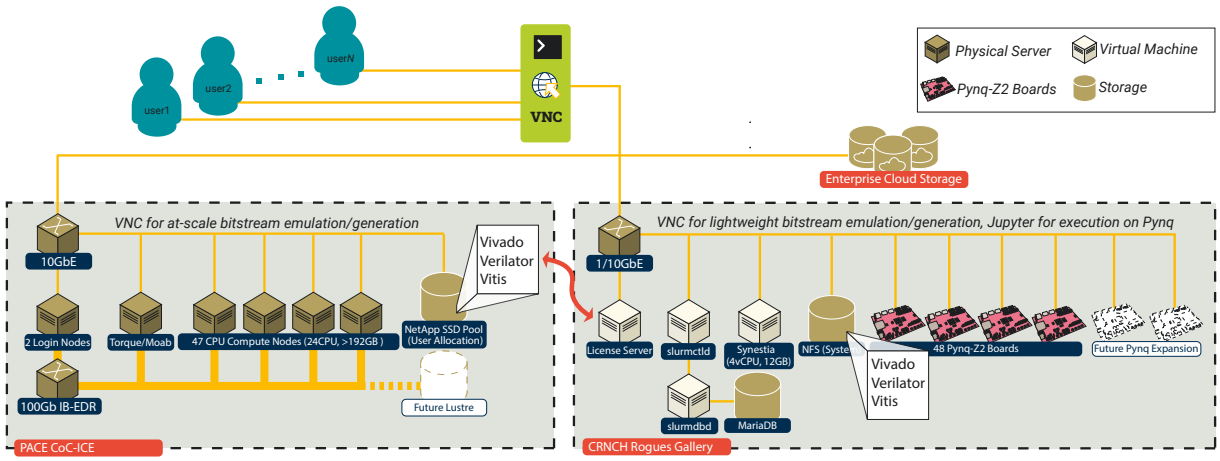

Figure 2: ICE and CRNCH RG Clusters

they create a program using software on their laptop and then program the device using a USB connection. In fact, related FPGA clusters [3, 9, 18, 31] have recently focused on supporting these types of efforts using cameras and web interfaces for the hands-on portions of the boards. However, our goals for this infrastructure are mostly focused on supporting a larger number of students using scalable, community-driven tools and allowing them to interact with the devices using supported interfaces like Jupyter notebooks with a low barrier to entry.

These goals for an FPGA-based cluster are further stretched by rapid growth in our undergraduate and graduate education enrollment and supply chain issues that have meant that students cannot actually purchase the required number of boards for a single course. This leads us to the the following key challenges that we looked to address:

- How can we support larger numbers of students for a novel FPGA cluster that ties in with Georgia Tech's existing Instructional Cluster Environment (ICE) to support educational objectives?
- How do we support key requirements like data separation and privacy for student data while allowing for free flow of data between isolated clusters and local access points?
- What kind of training would students require to migrate from the "hands-on" infrastructure to a remotely scheduled cluster that uses a scheduler like Torque/Moab or Slurm?

This work describes our approach to answer these questions and offers our insights into the infrastructure design choices that worked well and which obstacles proved challenging to overcome. We also describe some student response and feedback that we anticipate using to improve the system configuration and user workflows over time. The described infrastructure will also be shared in an open-source fashion so that others may implement their own novel architecture clusters based on similar concepts.

2 CLUSTER OVERVIEW - ICE AND CRNCH RG

Figure 2 shows the layout of the two clusters that were used in Spring 2022 to support reconfigurable computing and computer architecture classes.

The PYNQ cluster is part of the NSF-funded novel architecture testbed, the Rogues Gallery [34], a center-based testbed that is focused on near-term "post-Moore" computing including next-generation HPC, neuromorphic, near-memory, and reversible computing amongst other topics. The PIs of this Center for Research into Novel Computing Hierarchies (CRNCH) work closely with Georgia Tech's high-performance computing organization, the Partnership for an Advanced Computing Environment (PACE). In a sense, the Rogues Gallery helps to prototype small, next-generation systems while PACE enables the large-scale deployment of cutting-edge HPC and HTC environments that includes the NSF funded Hive project [20] and Buzzard project [2], as well as the Phoenix cluster [11] that ranked #277 on the Top500 November 2020 list.

In addition to the aforementioned research computing clusters, PACE also supports two instructional clusters, CoC-ICE and PACE-ICE [1]. CoC-ICE is a dedicated instructional HPC cluster administered as a collaboration between the College of Computing (CoC) and PACE and hosts courses in computing and, as appropriate, electrical engineering. Comparatively, PACE-ICE supports courses that teach and use scientific computing in engineering, physical and social sciences, and humanities and arts, plus training workshops in research computing offered by PACE to the entire campus community. Automated tools retrieve course registration lists from the registrar and provision student access and storage allocations. The heterogeneous CoC-ICE cluster is comprised of 45 nodes featuring Intel's Cascade Lake processors, and the cluster hosts 36 Tesla V100 and 24 RTX6000 Nvidia GPUs. All nodes are connected via a 100 Gbps InfiniBand network to support parallel computing. A NetApp device offers network storage to all students and instructors, as well as shared space for instructors to distribute course materials or install custom software. As a part of the PACE ecosystem, CoC-ICE provides the full suite of scientific software and tools available through the PACE software stack and trains

students in using an HPC cluster, including submitting jobs to a scheduler (Moab/Torque), loading software modules with lmod, and managing network and local storage.

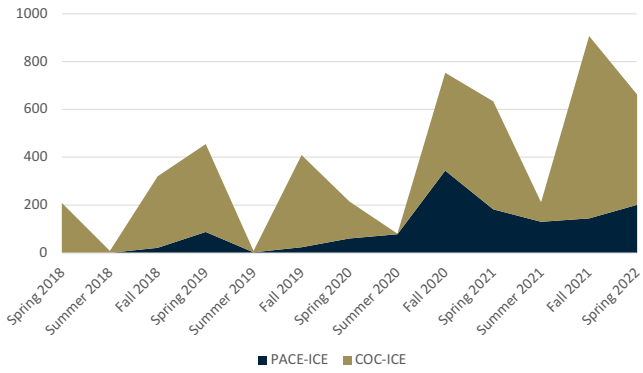

Figure 3: The CoC-ICE and PACE-ICE clusters have enhanced education for an increasing number of students since inception. The graph shows the number of students who ran jobs on each cluster during each semester, based on data from PACE's Open XDMoD [25] instance.

The instructional clusters have received significant demand and use from students and faculty. In 2021, the two ICE clusters hosted 20+ courses per semester and 60+ workshops, enabling over 1600 students to submit about 220,000 jobs and consume over 550,000 CPU hours. Since the launch of ICE in 2018, the clusters have supported course access for over 4,000 GT students. Figure 3 provides the breakdown of active students per semester, which is derived from the data presented by Open XDMoD [25].

More recently, the ICE clusters have seen an increase in demand for resources to support less traditional workflows and computational disciplines. Support requests from faculty include CPU/GPU heterogeneity for architecture-specific algorithm design, OpenGL-capable GPUs for simulation visualizations, and cloud-facing application interfaces to support hybrid activities, to name a few. Additionally, interest in ICE access for courses representing the so-called "long tail of science" [17] has been increasing. For example, ICE resources were utilized for "Data Analytics and Security" from the School of International Affairs and "Computational Musicology" from the School of Music, as highlighted in the Fall 2021 issue of the PACE Newsletter [21].

Through PACE's facilitation of courses on the ICE clusters and direct feedback from faculty teaching the courses, the major challenges that PACE observed and learned about involve preparing undergraduate students. Many students have little or no prior computing experience, especially on an HPC cluster, leading to issues with code editing and compilation, SSH access to and navigation within the ICE clusters, and batch submission system entry to compute resources. In working with faculty on facilitating their courses on ICE clusters, some have expressed that at least 50% of the in-class time is spent troubleshooting technical computing issues rather than focusing on the interesting scientific problems and applications.

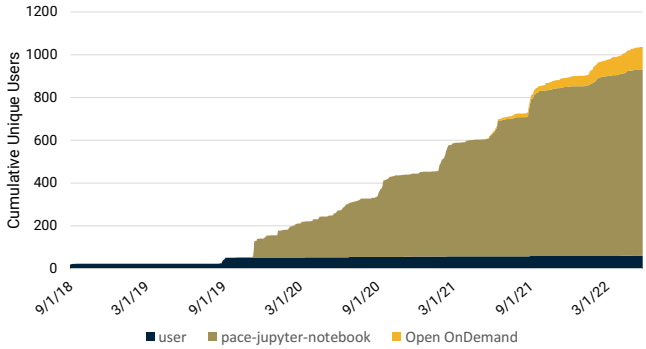

Figure 4: The deployment of PACE's Jupyter notebook wrapper in Fall 2019 eased access to Jupyter notebooks on PACE clusters and quickly achieved high utilization. Its use has continued to grow, and PACE's recent beta testing of Open OnDemand has added to the number of students and researchers using Jupyter notebooks to perform calculations on PACE clusters.

These challenges have driven the adoption of user-focused tools for interactive computing (e.g., Jupyter Notebook) as well as PACE-developed wrapper scripts for workflow abstraction [27, 28]. In late 2019, PACE deployed this prototype wrapper service across all PACE resources to make Jupyter Notebooks and VNC more accessible. This effort led to a rapid spike in the usage of Jupyter notebooks across resources as shown in Figure 4, where the number of unique users and the frequency of Jupyter notebook use quickly increased each month to several hundred unique users per month. Furthermore, these efforts set the stage for subsequent initiatives to continue lowering the barrier to entry and further democratize access to advanced research computing resources, such as the Apache Airavata [26] based Hive science gateway [19], and multiple Open OnDemand [8] instances for PACE clusters [22–24]. This has led to a strong collaboration with CRNCH's Rogues Gallery in deploying similar interactive computing wrappers and eventually Open OnDemand on the PYNQ FPGA and Arm-based Octavius [10] clusters.

3 CLASS REQUIREMENTS FOR NOVEL ARCHITECTURES

In contrast to the CPU and GPU hardware hosted in the ICE environments, novel architecture ecosystems used for class present challenges in scalability and resilience. In particular, FPGAs do not inherently provide the layer of indirection supported by CPU servers that is needed to allow multiple users to simultaneously utilize the underlying hardware resources; instead, this functionality must be setup by administrators as described in Section 4. Without this indirection, a separate FPGA board with power and networking would be required for each student. It is painfully apparent on how quickly this methodology becomes unscalable.

The two classes that were targeted for integration into the combined ICE and CRNCH environments were an undergraduate CS class built around processor design (CS 3220) [15] and a graduate

ECE class focused on the fundamentals of parallel programming for FPGAs (ECE 8893) [7]. These two classes had an enrollment of 90 students and 45 students respectively.

The primary objective of CS 3220 is to teach undergraduate computer science students the fundamentals of logic design and FPGAs by designing a 5-stage processor pipeline. These students achieve this goal by writing Verilog code, simulating their designs using Verilator [30] and Vivado [33] tools and then by generating a bitstream that is targeted for the PYNQ Z2 FPGA [29] platform. To achieve this goal, we needed a cluster of PYNQ FPGA boards that could be networked and managed without user intervention and that could give the end user the perception that they had full control over a single FPGA board.

This processor design class utilized the PYNQ cluster for two homeworks (10 and 11) and class projects (4 and 5). Project 4 [13] involved students adding branch prediction with a Branch Target Buffer (BTB) to their pipeline design and then generating a wrapper for their design. This wrapper would allow their designed processor pipeline to interface a programmable logic (PL) design with the processor side (PS) of the PYNQ FPGA. Students would then generate the bitstream and use a Jupyter Notebook to interface with the bitstream and produce a result. Project 5 [14] involved students creating a edge filter with High-Level Synthesis (HLS) and performing optimizations to improve the design latency and resource utilization. Students also generated their own intellectual property (IP) using Vitis HLS, imported it into a Vivado project, generated the bitstream, and verified results on the PYNQ-Z2 boards. Homeworks 10 and 11 [12] provided supporting material and background to achieve the steps needed for successful completion of projects 4 and 5 including accessing the remote PYNQ FPGA cluster, bitstream generation, and Vitis HLS.

For ECE 8893, the primary objective is to teach graduate students the basic architecture of FPGAs and System-on-Chips (SoCs), HLS programming and algorithm design considerations, and application opportunities for FPGAs. These students leveraged the Xilinx Vivado tools and Vitis HLS [32] to map C code to FPGA-based IP and accelerators. As with CS 3220, the cluster of PYNQ FPGA boards was used to provide students with a managed device that could run the same workflows as if the user had access to the physical hardware.

Graduate students in ECE 8893 utilized the PYNQ cluster for lab 3 [6], which was comprised of two tasks to familiarize students with the hardware and workflow. The first task asked students to read a randomized array from DRAM, increment each element by 1, and write it back to DRAM. The second task required students to implement a portion of their final project (or lab 2 code if their final project did not include HLS programming). Both tasks required submission of the host code, bitstream, and hardware description file. The second task required a report summarizing the correctness of the result and a comparison of the execution time and the HLS synthesis report prediction.

4 DESIGNING A NEW STUDENT WORKFLOW FOR FPGAS

Traditionally, Xilinx PYNQ boards are single user devices that run a static Jupyter server as the root user, which provides the ability

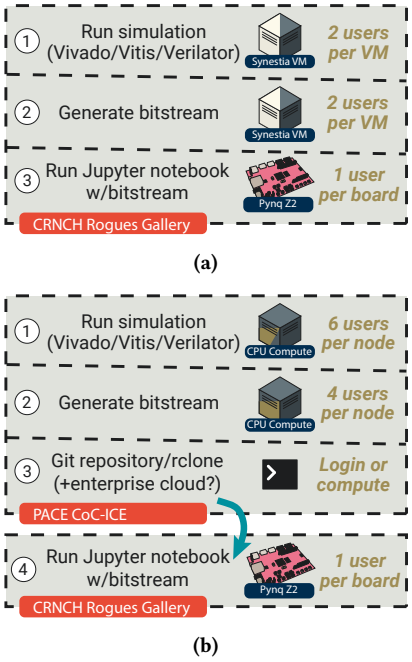

Figure 5: (a) Workflow on CRNCH Rogues Gallery only, with Synestia VMs and PYNQ-Z2 boards. (b) Workflow using PACE CoC-ICE compute nodes and CRNCH Rogues Gallery PYNQ-Z2 boards.

to leverage custom bitstreams and overlays through the Python interface. Typically the student runs as a sudo user, *xilinx*, which allows them to run the notebooks and program bitstreams directly into the programmable logic (PL) part of the board.

Figure 5a shows the design of our previous workflow from earlier semesters where students used a tool like x2go to connect to simulation and bitstream generation VMs in the *synestia* cluster. Then students would ssh or login directly to a PYNQ board as the *xilinx* user and load their bitstream from a network shared folder.

As previously discussed, this VM setup was not feasible to scale up for over 100 students, and, due to supply chain limitations, there was no option for students to buy or borrow their own board for local usage. For these reasons, we focused on the development and integration of the workflow in Figure 5b where students used CoC-ICE for simulation and bitstream generation and the Slurm-enabled PYNQ cluster for on-board testing and debugging.

4.1 Filesystem considerations

As a student-focused project, this effort had several considerations related to student data and privacy that needed to be considered. Specifically, we cannot explicitly share data between the CoC-ICE file server partition and our CRNCH testbed.

The CoC-ICE filesystem is hosted on a NetApp device, providing each student with a 15 GB quota and home directory. Daily snapshots produce backups, and each student's directory is accessible only to that student for privacy. Students on the cluster are not

associated with a particular course, only a school, in order to avoid revealing sensitive course enrollment information.

On the CRNCH-hosted cluster (*synestia* and the PYNQ boards), we created a single folder per class to allow students to have a consistent nethome on the login node and on the target PYNQ devices. This setup also allows for easily "resetting" the student nethome between semesters to preserve student privacy.

These two decisions do introduce one additional requirement for students since the two clusters do not share the same data sets. As show in Figure 5b, students need to save their bitstream to a Git repo on CoC-ICE and then pull it onto the CRNCH cluster before proceeding to testing with physical hardware. The instructors and TAs of the course provided information on using git for this small transfer step as well as how to use rclone to save files to a student-owned Box or Dropbox folder, which are hosted via Georgia Tech's enterprise cloud services for students.

4.2 PYNQ board setup

The PYNQ boards run a 32-bit Arm-based version of Ubuntu 20.04 (PYNQ 2.7 image for Z-2 boards) while the hosting *synestia* VMs use 64-bit Ubuntu 18.04. CoC-ICE servers currently run Red Hat Enterprise Linux 7.6.

To deploy Slurm and appropriate settings for the PYNQ boards, we used Ansible scripts and hand-built deb files for Slurm 21.08 for both the PYNQ nodes and the Synestia VM. For the exam data collection in Section 6, a common Slurm database daemon (DBD) was used that is hosted to track and account for all CRNCH Rogues Gallery resources.

We also modified the local sudoers file to allow normal users to run the Jupyter notebook as a root user without a password. This tweak was made to allow users to both access bitstreams they generated and to program them on the local PYNQ board.

4.3 Adoption of PACE wrapper scripts for PYNQ cluster

We adopted and modified the PACE wrapper script for Jupyter notebooks on the CoC-ICE cluster to support Slurm in place of Torque/Moab that the PACE clusters currently utilize. More specifically, we create an sbatch script that users call from a wrapper script that is shown in Listing 1. This script creates a Jupyter notebook instance on the PYNQ node and then returns the port forwarding information that a user can use to forward the remote port 8888 to a randomized local port that is then forwarded over an SSH tunnel.

Listing 2 shows the output of running this sbatch script from the user's perspective. The user launches a new Slurm job with a simple script, and they are then directed to open a Jupyter notebook at port 58786 in their local web browswer.

In addition to scripting changes, we updated the Slurm prolog and epilog to link a user's nethome directory into the /root directory and then unlink it at the end of the job. This allowed students to run the job script, log into a notebook as a root user, and run bitstreams from their nethome directory.

5 DEPLOYMENT AND DEBUGGING

We deployed the infrastructure for the two courses and collected feedback from students throughout the two months when these

```
cd $SLURM_SUBMIT_DIR
PIPEFILE=${1}
rm ${PIPEFILE}
mkfifo ${PIPEFILE}
(jupyter notebook --no-browser --ip=${HOSTNAME} --port=8888
↪  2>&1 | tee ${PIPEFILE} ) &
while read -r line
do
    echo "$line"
    if [[ "$line" == "http://pynq-z2-"* ]]
    then
        HOST=$(sed -e 's#http\://\([^:]*\)\:.*#\1#'<<<$line)
        PORT=$(sed -e
        ↪  's#http\://[^:]\:\([0-9]*\)/?.*#\1#'<<<$line)
        TOKEN=$(sed -e 's#.*/?\(.*\)#\1#'<<<$line)
        break
    fi
done < ${PIPEFILE}
echo "Host: $HOST"
echo "Port :$PORT"
echo "Token: $TOKEN"
```

Listing 1: Slurm Jupyter Notebook Script

```
Submitting job via sbatch slurm-jupyter-notebook.batch...
Submitted batch job 14

Job successfully submitted!
Waiting for job to start

Starting jupyter notebook...

Connect to your jupyter notebook via the following steps:
    1) Press SHIFT + ~ then SHIFT + C to open an SSH console
    ↪  (The prompt 'ssh>' should appear on the next line)
       ***Note: '~' MUST be the first character on the line
       ↪  to be recognized as the escape character,in which
       ↪  case it will not appear on your terminal.***
       ***If you see the '~' character when you start typing,
       ↪  delete it, hint 'ENTER'and type 'SHIFT' + '~' +
       ↪  'C' again.***
    2) Type -L 58786:pynq-z2-42:58786 and then ENTER
    3) Connect your browser to http://localhost:58786/ and
    ↪  enter 'xilinx' in the password prompt.
```

Listing 2: Slurm Notebook Script Output

classes were most active. With the feedback from earlier homeworks and projects, we improved the cluster robustness and resolved issues including incompatible bitstreams and too many active user sessions. Most students in CS 3220 and ECE 8893 were able to verify their design on the cluster via homework and projects. We were also able to support an exam where 41 boards were available and 35 students actively working on the cluster during a 3-hour time period.

A commonly encountered issue was getting *[Error 110] Connection time out error* when loading a bitstream through the PYNQ overlay library. The TAs for 3220 helped to confirm that the root cause of this issue was using a very similar but slightly incompatible bitstream generated for another board that then broke the board logic. We also noticed that once a board was programmed with an incompatible bitstream, the board remained dysfunctional until it was rebooted. To resolve this issue, we experimented with using a cron script and Slurm epilog scripts to reboot the board at the end of every user session. The next generation of this infrastructure will allow for remote power cycling of these boards, which should also help to mitigate bad bitstreams by using cold reboots of the hardware.

We also observed that when there were no extra board resources available, a request to allocate new resources would sometimes fail with an ssh hostname error. We attempted to mitigate this error by providing more detailed feedback on failures in the wrapper scripts used to launch jobs.

At the beginning of the FPGA cluster deployment, we informed the students about the basics of Slurm infrastructure so that they understood how the cluster allocated resources and were able to monitor and manage their session. However, we still encountered situations where students would attempt to allocate multiple boards when a wrapper script initially failed to launch a job. To prevent this from happening, we asked students to check the status of their sessions with commands like *squeue* and to terminate any failed or idle jobs. This guidance worked well for students with a background in HPC systems, but a minority of students still struggled with using the new Slurm-based infrastructure. Ultimately, we determined the best course for future semesters is to provide more detailed Slurm training and examples as well as to set explicit job limits for students using Slurm accounting.

6 STUDENT OUTCOMES AND SENTIMENTS

While this infrastructure is brand new and should still be considered "beta" due to its accelerated deployment, we were able to pull a few statistics from the undergraduate course's final exam period using SlurmDBD statistics, and we report on student sentiment on the hardware based on a post-course survey.

6.1 Slurm Reporting for Final Exam

Slurm accounting information was collected in the Slurm database over the time frame of 4 hours (from 2pm-6pm) for a final exam. In this time frame, 132 jobs were started for 36 unique users. Jobs started in this 4-hour window ran for a mean of 40.72 minutes (sd = 35.17 minutes). At peak job workload during the time frame for the final exam, 36 jobs ran for 25 concurrent users (at 4:33pm). Table 1 demonstrates how a final exam can be a "peak workload" scenario using this PYNQ cluster.

Time	Unique Users	Jobs
02:00pm	0	0
02:30pm	3	3
03:00pm	11	12
03:30pm	18	24
04:00pm	19	24
04:30pm	25	33
05:00pm	25	34
05:30pm	18	24
06:00pm	13	16

Table 1: Number of Unique Users and Jobs During Final Exam (2-6pm)

Eight students had more than two jobs running at once that we needed to limit manually in order to provide fair access to available FPGA resources. This points to a need for priority reservation for exam participants and also stricter limits on the maximum number of jobs run as enforced by Slurm accounting.

6.2 Survey Results

The survey results shown in Figure 6 demonstrate some of the challenges and opportunities of this cluster and the related integration of educational workflows. Students appreciated not needing to buy boards, but they also expressed frustration at needing to understand Slurm and PYNQ-related errors that had not typically been encountered in previous semesters.

Our main feedback from this survey is a positive one. We believe that this workflow is what students would like to engage with as long as we can improve the scripting and documentation to better support the dual workflow of simulation and synthesis with CoC-ICE and the physical execution on CRNCH RG FPGAs.

7 LESSONS LEARNED

Despite the short timeline for this deployment, we learned several lessons that will be useful for future joint workflow deployments:

(1) Each novel architecture typically has a workflow that involves simulation or emulation, and we can utilize traditional HPC via clusters like CoC-ICE and PACE-ICE to help accelerate this phase of the workflow for larger numbers of students.
(2) Data separation and scheduling constraints become more complex when student data is involved, but we can still use best practices from one cluster to help stand up and measure data from a novel architecture platform like the CRNCH RG PYNQ cluster.
(3) Undergraduate students typically struggle with learning new scheduling concepts that they have not encountered before in their academic career. One possible approach is to provide a standardized scheduling environment (i.e., just support one of Slurm or Torque/Moab), create examples and wrapper scripts for students, and simplify their workflow as much as possible when multiple clusters are used.

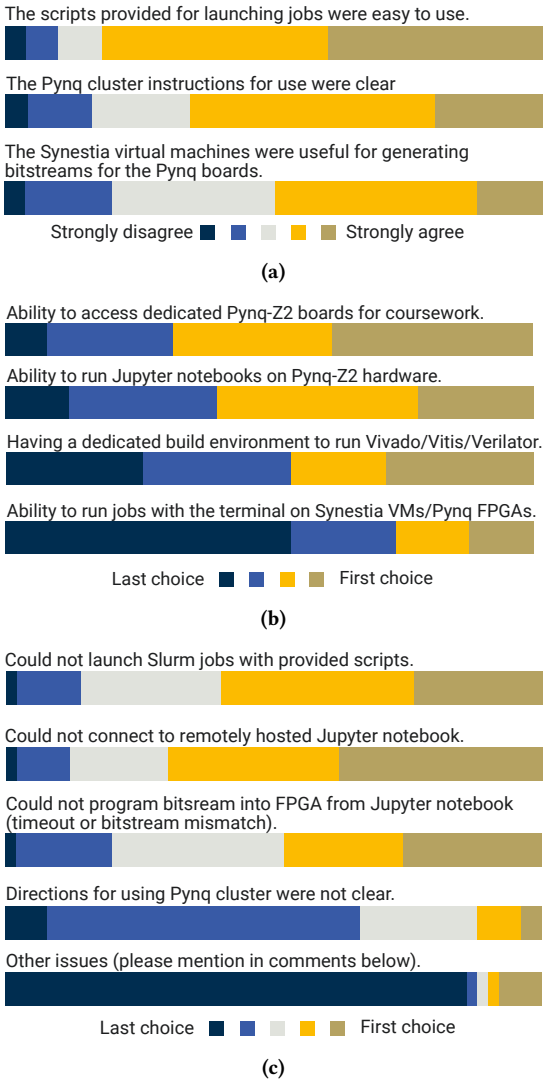

Figure 6: Student rankings of cluster (a) benefits (b) features and (c) challenges.

8 RELATED WORK

While we believe that this particular implementation of a PYNQ-based FPGA cluster is unique due to its focus on interactive scalability, there are several other research- and educational-oriented clusters that have focused specifically on FPGAs.

At a vendor level, Xilinx supports its Heterogeneous Accelerated Compute Clusters (HACC/XACC) at universities in the US and abroad, and Intel provides its DevCloud for remote access by researchers. It is notable that both of these infrastructures are set up for small numbers of users to join at once and may not be suitable for large, transient student populations. ECELabs.io [16] is a recent cluster that provides students with a "visual" representation of devboards like PYNQ and Nexys FPGAs where students can remotely toggle the buttons on an actual board instance and see the LED output from the board. As mentioned earlier, there are also several interesting FPGA cluster efforts [3, 9, 18, 31] that are focused on supporting cameras and web-based interfaces for hands-on usage of these small devices. These efforts are not mutually exclusive to the Slurm- and Jupyter-based workflow presented here, and our hope is that our public documentation of this work will allow for the wider adoption of both hands-on and scalable approaches for these clusters.

In addition to these vendor efforts, there are many interesting research projects focused on using larger FPGA boards like Xilinx Alveo and Intel's Arria 10 to build larger clusters, including efforts to support data analytics with DASK, DASK on Alveo boards with some PYNQ scaling, and the GreenFlash FPGA cluster [4] for providing real-time control of adaptive optics for the Extremely Large Telescope (ELT). Testbeds focused on using Alveo boards include the Open Cloud Testbed run by Northeastern University [5].

9 CONCLUSIONS

This effort demonstrates the power of traditional HPC along with adaptations for future novel architectures including FPGAs.

From this work, we expect that several follow-on efforts will continue to improve both CoC-ICE and the CRNCH testbed, including combined support for Slurm and integration of Slurm accounting data with PACE's Open XDMoD [25] instance for metrics collection and monitoring. Deeper integration with XDMoD would allow us to better measure the impact of Pynq cluster. We do need better scripts to help synchronize Slurm accounting and LDAP groups, but this would likely be a joint concern for both clusters.

Secondly, we believe that the addition of new services like Open OnDemand on CoC-ICE and the CRNCH testbed will likely simplify the job launch process since it eliminates the need for users to use and understand SSH port forwarding. As we grow CoC-ICE to 75 nodes in the upcoming academic year, we anticipate that both clusters will benefit from continued joint efforts to simplify and standardize workflows for both traditional HPC and novel architectures.

ACKNOWLEDGEMENT

This research was supported by the NSF MRI award #1828187: "MRI: Acquisition of an HPC System for Data-Driven Discovery in Computational Astrophysics, Biology, Chemistry, and Materials Science." Additionally, this research was supported in part through research infrastructure and services provided by the Rogues Gallery testbed hosted by the Center for Research into Novel Computing Hierarchies (CRNCH) at Georgia Tech. The Rogues Gallery testbed is primarily supported by the National Science Foundation (NSF) under NSF Award Number #2016701. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s), and do not necessarily reflect those of the NSF.

REFERENCES

[1] Mehmet Belgin, Trever C. Nightingale, David A. Mercer, Fang Cherry Liu, Peter Wan, Andre C. McNeill, Ruben Lara, Paul Manno, and Neil Bright. 2018. ICE: A Federated Instructional Cluster Environment for Georgia Tech. In *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*. Association for Computing Machinery, New York, NY, USA, Article 16, 7 pages. https://doi.org/10.1145/3219104.3219112

[2] Mehmet Belgin, Semir Sarajlic, Ruben Lara, Laura Cadonati, Nepomuk Otte, Ignacio J. Taboada, Gregory L. Beyer, Norman B. Bonner, Michael Brandon, Pam Buffington, J. Eric Coulter, Aaron Jezghani, David Leonard, Fang (Cherry) Liu, Paul D. Manno, Craig A. Moseley, Trever C. Nightingale, Ronald Rahaman, Kenneth J. Suda, Peter Wan, Michael D. Weiner, Deirdre Womack, Dan Zhou, Marian Zvada, Andre C. McNeill, Neil C. Bright, Robert W. Gardner, Paschalis Paschos, Lincoln Andrew Bryant, Judith Lorraine Stephen, James Alexander Clark, Peter F. Couvares, Brian Hua Lin, Todd Tannenbaum, and Gregory Thain. 2022. Buzzard: Georgia Tech's Foray into the Open Science Grid. In *PEARC22: Practice and Experience in Advanced Research Computing 22*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3491418.3535135

[3] Vassilis Fotopoulos, Anastasios Fanariotis, Theofanis Orphanoudakis, and Athanassios N. Skodras. 2015. Remote FPGA Laboratory Course Development Based on an Open Multimodal Laboratory Facility. In *Proceedings of the 19th Panhellenic Conference on Informatics (PCI '15)*. Association for Computing Machinery, New York, NY, USA, 447–452. https://doi.org/10.1145/2801948.2801950

[4] D Gratadour, M Andrighettoni, G Angerer, A Basden, J Bernard, R Biasi, U Bitenc, T Buey, R Dembet, H Deneux, et al. 2017. Green Flash: Exploiting future and emerging computing technologies for AO RTC at ELT scale. In *AO4ELT5, http://research. iac. es/congreso/AO4ELT5/pages/proceeding0108. html.*

[5] S Handagal, Martin Herbordt, and Miriam Leeser. 2021. OCT: The Open Cloud FPGA Testbed. In *31st International Conference on Field Programmable Logic and Applications (FPL)*.

[6] Cong (Callie) Hao. 2022. ECE 8893 Lab 3 Description. https://github.com/sharc-lab/FPGA_ECE8893/tree/main/2022_Spring/Lab3

[7] Cong (Callie) Hao. 2022. Georgia Tech ECE 8893 Course Page (webpage). https://github.com/sharc-lab/FPGA_ECE8893

[8] David E. Hudak, Douglas Johnson, Jeremy Nicklas, Eric Franz, Brian McMichael, and Basil Gohar. 2016. Open OnDemand: Transforming Computational Science Through Omnidisciplinary Software Cyberinfrastructure. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale (XSEDE16)*. Association for Computing Machinery, New York, NY, USA, Article 43, 7 pages. https://doi.org/10.1145/2949550.2949644

[9] Rania Hussein and Denise Wilson. 2021. Remote Versus In-hand Hardware Laboratory in Digital Circuits Courses Remote versus In-Hand Hardware Laboratory in Digital Circuits Courses.

[10] Aaron Jezghani, Kevin Manalo, Will Powell, Jeffrey Valdez, and Jeffrey Young. 2022. Onboarding Users to A64FX via Open OnDemand. In *International Conference on High Performance Computing in Asia-Pacific Region Workshops*. 78–83.

[11] Aaron Jezghani, Semir Sarajlic, Michael Brandon, Neil Bright, Mehmet Belgin, Gregory Beyer, Christopher Blanton, Pam Buffington, J. Eric Coulter, Ruben Lara, Lew Lefton, David Leonard, Fang (Cherry) Liu, Kevin Manalo, Paul Manno, Craig Moseley, Trever Nightingale, N. Bray Bonner, Ronald Rahaman, Christopher Stone, Kenneth J. Suda, Peter Wan, Michael D. Weiner, Deirdre Womack, Nuyun Zhang, and Dan Zhou. 2022. Phoenix: The Revival of Research Computing and the Launch of the New Cost Model at Georgia Tech. In *PEARC22: Practice and Experience in Advanced Research Computing 22*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3491418.3530767

[12] Hyesoon Kim. 2022. CS 3220 Homework 10+11 Description. https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/hw10.md

[13] Hyesoon Kim. 2022. CS 3220 Project 4 Description. https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/project4.md

[14] Hyesoon Kim. 2022. CS 3220 Project 5 Description. https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/project5.md

[15] Hyesoon Kim, Bhanu Garg, Anurag Kar, and Xueyang Liu. 2022. Georgia Tech CS3220 Course Page (webpage). https://gt-cs3220.github.io/

[16] Junfei Li. 2021. ECELabs.io website. https://www.ecelabs.io

[17] Rob Mitchum. 2012. Unwinding the 'Long Tail' of Science. https://www.ci.uchicago.edu/blog/unwinding-long-tail-science

[18] Abd El-Rahman Mohsen, Mohamed Youssef GadAlrab, Zeina elhaya Mahmoud, Gameel Alshaer, Mahmoud Asy, and Hassan Mostafa. 2019. Remote FPGA Lab For ZYNQ and Virtex-7 Kits. In *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*. 185–188. https://doi.org/10.1109/MWSCAS.2019.8885064

[19] PACE. 2021. Georgia Tech Hive Gateway. https://gateway.hive.pace.gatech.edu Accessed: 2022-05-29.

[20] PACE. 2021. Georgia Tech Hive project. https://pace.gatech.edu/new-hpc-cluster-hive

[21] PACE. 2021. PACE Newsletter - Fall 2021. https://pace.gatech.edu/sites/default/files/pace-newsletter-fall2021.pdf. Accessed: 2022-05-27.

[22] PACE. 2022. Hive Open OnDemand. https://ondemand-hive.pace.gatech.edu/. Accessed: 2022-05-29.

[23] PACE. 2022. PACE-ICE (Beta) Open OnDemand. https://ondemand-pace-ice.pace.gatech.edu/. Accessed: 2022-05-29.

[24] PACE. 2022. Phoenix Open OnDemand. https://ondemand-phoenix.pace.gatech.edu/. Accessed: 2022-05-29.

[25] Jeffrey T. Palmer, Steven M. Gallo, Thomas R. Furlani, Matthew D. Jones, Robert L. DeLeon, Joseph P. White, Nikolay Simakov, Abani K. Patra, Jeanette Sperhac, Thomas Yearke, Ryan Rathsam, Martins Innus, Cynthia D. Cornelius, James C. Browne, William L. Barth, and Richard T. Evans. 2015. Open XDMoD: A Tool for the Comprehensive Management of High-Performance Computing Resources. *Computing in Science Engineering* 17, 4 (2015), 52–62. https://doi.org/10.1109/MCSE.2015.68

[26] Marlon Pierce, Suresh Marru, Lahiru Gunathilake, Thejaka Amila Kanewala, Raminder Singh, Saminda Wijeratne, Chathuri Wimalasena, Chathura Herath, Eran Chinthaka, Chris Mattmann, Aleksander Slominski, and Patanachai Tangchaisin. 2014. Apache Airavata: Design and Directions of a Science Gateway Framework. In *2014 6th International Workshop on Science Gateways*. 48–54. https://doi.org/10.1109/IWSG.2014.15

[27] Semir Sarajlic, Neranjan Edirisinghe, Yuriy Lukinov, Michael Walters, Brock Davis, and Gregori Faroux. 2016. Orion: Discovery Environment for HPC Research and Bridging XSEDE Resources. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale (XSEDE16)*. Association for Computing Machinery, New York, NY, USA, Article 54, 5 pages. https://doi.org/10.1145/2949550.2952770

[28] Semir Sarajlic, Neranjan Edirisinghe, Yubao Wu, Yi Jiang, and Gregori Faroux. 2017. Training-Based Workforce Development in Advanced Computing for Research and Education (ACoRE). In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact (PEARC17)*. Association for Computing Machinery, New York, NY, USA, Article 71, 4 pages. https://doi.org/10.1145/3093338.3104178

[29] TUL. 2022. Pynq-Z2 FPGA. https://www.tulembedded.com/FPGA/ProductsPYNQ-Z2.html

[30] Veripool. 2022. Verilator. https://www.veripool.org/verilator/

[31] Marco Winzker and Andrea Schwandt. 2019. Open Education Teaching Unit for Low-Power Design and FPGA Image Processing. In *2019 IEEE Frontiers in Education Conference (FIE)*. 1–9. https://doi.org/10.1109/FIE43999.2019.9028694

[32] Xilinx. 2022. Vitis. https://www.xilinx.com/products/design-tools/vitis/vitis-platform.html

[33] Xilinx. 2022. Vivado. https://www.xilinx.com/products/design-tools/vivado.html

[34] Jeffrey S. Young, Jason Riedy, Thomas M. Conte, Vivek Sarkar, Prasanth Chatarasi, and Sriseshan Srikanth. 2019. Experimental Insights from the Rogues Gallery. In *2019 IEEE International Conference on Rebooting Computing (ICRC)*. 1–8. https://doi.org/10.1109/ICRC.2019.8914707