

---

# **Crnch Rogues Gallery Documentation**

***Release 1.0***

**Crnch Rogues Gallery**

**Sep 18, 2023**



## GENERAL TOPICS

<b>1</b>	<b>What is the Rogues Gallery?</b>	<b>3</b>
<b>2</b>	<b>Getting Started with Rogues Gallery</b>	<b>5</b>
<b>3</b>	<b>Rogues Gallery Hardware</b>	<b>9</b>
<b>4</b>	<b>Rogues Gallery Filesystems</b>	<b>13</b>
<b>5</b>	<b>Mailing Lists and Requesting Help</b>	<b>15</b>
<b>6</b>	<b>Vendor Forums</b>	<b>19</b>
<b>7</b>	<b>Contributing to this Documentation</b>	<b>21</b>
<b>8</b>	<b>Testbed Release Notes</b>	<b>25</b>
<b>9</b>	<b>RG Documentation Authors</b>	<b>27</b>
<b>10</b>	<b>Using Slurm with RG Systems</b>	<b>29</b>
<b>11</b>	<b>Open OnDemand</b>	<b>33</b>
<b>12</b>	<b>Visual Studio Code</b>	<b>43</b>
<b>13</b>	<b>Python Environments</b>	<b>47</b>
<b>14</b>	<b>Jupyter Notebooks</b>	<b>55</b>
<b>15</b>	<b>Using Scrontab with Slurm</b>	<b>59</b>
<b>16</b>	<b>Using GUI Applications with VNC</b>	<b>61</b>
<b>17</b>	<b>CI/CD Support</b>	<b>69</b>
<b>18</b>	<b>Rogues Gallery Workflows</b>	<b>71</b>
<b>19</b>	<b>Profilers</b>	<b>75</b>
<b>20</b>	<b>Instinct - AMD MI210</b>	<b>77</b>
<b>21</b>	<b>Quorra - NVIDIA Ampere GPUs</b>	<b>81</b>
<b>22</b>	<b>Violet - Sapphire Rapids</b>	<b>83</b>

<b>23</b>	<b>Octavius - A64FX Testbed</b>	<b>85</b>
<b>24</b>	<b>Kingpin - NVIDIA DevKit Systems</b>	<b>91</b>
<b>25</b>	<b>Lucata Pathfinder Getting Started</b>	<b>93</b>
<b>26</b>	<b>Lucata Pathfinder FAQs</b>	<b>97</b>
<b>27</b>	<b>Rudi2 - Jetson AGX Orin</b>	<b>99</b>
<b>28</b>	<b>Smart Networking - Getting Started</b>	<b>101</b>
<b>29</b>	<b>Bluefield-2 DPU</b>	<b>103</b>
<b>30</b>	<b>cuQuantum</b>	<b>105</b>
<b>31</b>	<b>Xilinx FPGAs - Getting Started</b>	<b>107</b>
<b>32</b>	<b>Xilinx Vivado Flow</b>	<b>111</b>
<b>33</b>	<b>PYNQ cluster</b>	<b>113</b>
<b>34</b>	<b>Xilinx Smart SSD</b>	<b>115</b>
<b>35</b>	<b>Vortex RISC-V GPGPU</b>	<b>117</b>
<b>36</b>	<b>Xilinx ML Tools</b>	<b>119</b>
<b>37</b>	<b>Intel OneAPI for Reconfigurable Computing</b>	<b>123</b>
<b>38</b>	<b>Using Intel OneAPI with CRNCH Rogues Gallery</b>	<b>127</b>
<b>39</b>	<b>FPGA Power Measurement</b>	<b>133</b>
<b>40</b>	<b>RISC-V Hardware</b>	<b>137</b>
<b>41</b>	<b>Frozone</b>	<b>139</b>
<b>42</b>	<b>Power Monitoring</b>	<b>141</b>
<b>43</b>	<b>CRNCH Rogues Gallery Tutorials</b>	<b>143</b>
<b>44</b>	<b>Near-memory Resources</b>	<b>145</b>
<b>45</b>	<b>GraphBLAS Resources</b>	<b>147</b>
<b>46</b>	<b>Neuromorphic Computing Resources</b>	<b>151</b>
<b>47</b>	<b>Quantum Computing Resources</b>	<b>155</b>
<b>48</b>	<b>Related Testbed Resources</b>	<b>159</b>



The Rogues Gallery is a new concept focused on developing our understanding of next-generation hardware with a focus on unorthodox and uncommon technologies. This project, initiated by Georgia Tech’s Center for Research into Novel Computing Hierarchies (CRNCH), will acquire new and unique hardware (ie, the aforementioned “rogues”) from vendors, research labs, and startups and make this hardware available to students, faculty, and industry collaborators within a managed data center environment. By exposing students and researchers to this set of unique hardware, we hope to foster cross-cutting discussions about hardware designs that will drive future performance improvements in computing long after the Moore’s Law era of “cheap transistors” ends.

To see what hardware the Rogues Gallery currently includes, please see our [hosted hardware page](#). For more information on the initial Rogues Gallery vision, please see our recent presentation at the 2021 CRNCH Summit [[Slides](#)] [[Talk](#)].

For updated status on the testbed please see our Spring 2022 talk [[Slides](#)]

**NSF Acknowledgment:** 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 documentation are those of the author(s), and do not necessarily reflect those of the NSF.

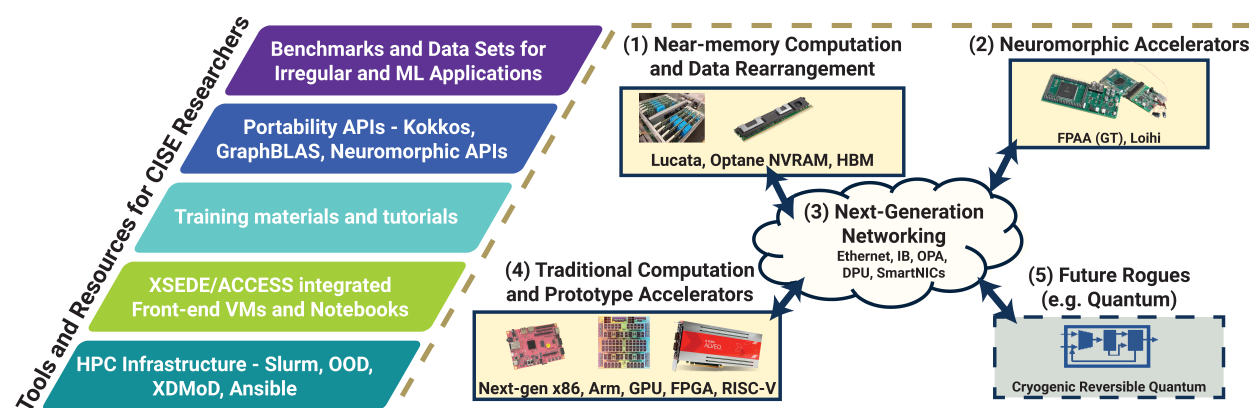


## WHAT IS THE ROGUES GALLERY?

The Rogues Gallery is a shared, open-access testbed run by the [Center for Research into Novel Computing Hierarchies \(CRNCH\)](#) at Georgia Tech. This testbed is focused on supporting “post-Moore” architectures, tools, and software. The RG testbed is currently funded by NSF’s Community CISE Research Infrastructure (CCRI) program as NSF award #2016701 starting in September 2020. The Rogues Gallery was started as a Georgia Tech specific testbed in 2017, and it has since grown to close to over 100 internal and external users.

### 1.1 What is in the Rogues Gallery?

Currently the Rogues Gallery has support for devices in the near-memory, traditional HPC, reconfigurable, and neuromorphic spaces. The CCRI grant will enable us to expand that to networking and quantum-related testbed equipment. The figure below shows a high-level overview of what the Rogues Gallery testbed looks like.



This testbed is meant to enable several CISE-specific research pillars including: 1) sparse and irregular HPC 2) hardware and software codesign 3) novel machine learning systems 4) and edge computing and next-generation networking.

### CISE Enabled Research Pillars

#### Sparse/Irregular HPC

- Graph analytics
- Scientific Computing
- Database and Big Data Acceleration



#### HW/SW Codesign

- Polyhedral compilation,
- Design of libraries, runtimes, APIs for novel devices
- Benchmarking and characterization

#### Machine Learning

- Low-power edge AI
- Autonomous vehicles
- SLAM for robotics
- Dynamic and life-long learning



#### Next-generation Networking

- 5G software stacks
- Edge computing services
- In-situ and encrypted data analysis
- Data reduction and line-speed DSP

### Rogues Gallery Hardware and Software Support

- Emu Pathfinder
- FPGA+HBM
- CASPIAN
- Tensor, Streaming Graph APIs
- ARM A64FX

- Emu Pathfinder
- FPGAs + RISC-V
- Optane
- Kokkos, Habana-C runtimes

- FPAA and CASPIAN
- EMU Pathfinder
- FPGAs
- RASP/TENNLab SW
- Emu Scikit-learn

- Ettus USRP-2947 and Ettus E-320
- Mellanox Bluefield NICs
- FPGAs
- FPAA and CASPIAN

## 1.2 Who are the users?

Anyone in the US can request an account for the CRNCH Rogues Gallery testbed with priority for CISE researchers, students, and industry partners.

## 1.3 How do I acknowledge the use of Rogues Gallery resources?

Please see our [project page](#) on citing the usage of the Rogues Gallery. This helps us acknowledge NSF and other support and assists in the growth of the testbed!

## GETTING STARTED WITH ROGUES GALLERY

### 2.1 How do I request an account on the Rogues Gallery (RG)?

Users should request a new account using our form on the CRNCH page [here](#). Note that it may take 2-3 business days for your account to be added after you are approved.

For both internal and external users, we ask for “an RG contact”. If you’ve talked to one of the key personnel for the RG testbed you can put this person down, or you can put down your research advisor if you are external to GT but working at a university.

For GT users, please put your research advisor or down one of the testbed personnel you have talked to. We will also need your GT (Passport) username to add login access for you.

**NOTE:** For GT faculty or staff looking to add new *external* users, it is very helpful to us if you can sponsor the users as [Passport guests](#). Then the above form can be filled out with the external user’s guest username which helps to streamline the access process. You can sponsor a guest account with Passport for up to one year and then renew the user if they are still an active collaborator.

### 2.2 How do I Access RG machines?

**NOTE:** You should use your GT Passport username (e.g, gburdell3) and password to access all CRNCH RG nodes and resources!

Once you receive a note that your account has been approved, you can test your login to the testbed. To access Rogues Gallery machines from off campus you need to either

- 1) SSH to the testbed via the login node from a terminal session.

OR

- 2) Use Open OnDemand for a GUI-based session. See [this page for more information on using Open OnDemand](#)

For SSH, you should use your Georgia Tech account username and password to log into all nodes using the gateway login node, `rg-login.crnch.gatech.edu`. `rg-login` is available from off the campus network and the VPN, and other nodes are accessible from the login node. As an example:

```
<yourmachine>$ ssh <your-gt-acctname>@rg-login.crnch.gatech.edu
rg-login.crnch.gatech.edu
```

```
=====
```

```
This is the main RG login node. From this node you can access all CRNCH resources. To
↪see a complete list of available hardware please visit: https://gt-crnch-rg.
```

(continues on next page)

(continued from previous page)

```
↪readthedocs.io/en/main/general/rg-hardware.html  
<your-gt-acctname>@#rg-login$
```

Once you’ve logged in, you can use Slurm to request other nodes within the testbed. See more information on Slurm at [this page](#).

Note that you can also use VSCode to log into the Rogues Gallery via its terminal functionality. See [this page for more details](#).

## 2.3 What machines are available in the Rogues Gallery?

Note that all machines have a **.crnch.gatech.edu** suffix and some examples include:

- **rg-login** login VM for off-campus access. Used as a gateway node
- **rg-emu-dev**: VM for Emu compilation and simulation
- **rg-fpga-dev-<1-4>**: VMs for FPGA compilation and simulation
- **rg-neuro-dev**: VM for neuromorphic tools
- Please see [this page](#) for a complete list of available machines in the RG testbed

## 2.4 VPN Requirements (for GT users only)

You can access all CRNCH resources using login nodes like *rg-login* or the Open OnDemand web interface. However, you can also use GT’s [VPN](#) solution as an alternative to access some VMs and nodes directly. Note that this may require that you set up two-factor authentication on your phone and link it with your account name. You shouldn’t need to use the VPN to access Rogues Gallery resources and materials, but for special requirements please email us at [crnch-rg-help@cc.gatech.edu](mailto:crnch-rg-help@cc.gatech.edu).

Other Georgia Tech resources like PACE’s Phoenix cluster require the usage of the VPN, and you can read more about these requirements at their [site](#)

## 2.5 How do I work with *novel architecture of interest X*?

First of all, check the sidebar for the appropriate architecture category and read through the existing documentation. Note that we are migrating some of our pages from our internal wiki to this public ReadTheDocs site over the next few months. You can use your GT username and password to log into the internal wiki.

Once you have done this, if you have questions please [ask on Teams](#) or [via help ticket](#).

- [Rogues Gallery Hardware Master List](#)
- [Lucata Pathfinder Getting Started](#)
- [Reconfigurable Computing HW and Tools](#)
  - [Intel FPGAs Getting Started](#)
  - [Xilinx FPGAs Getting Started](#)
- [Quantum Computing](#)
- [Neuromorphic Computing](#)

- Smart Networking
- RISC-V

## 2.6 What are some best practices for using the RG hardware?

1. Make sure to back your code up, preferably with a GT Github or external Github repository. While we do have a [shared filesystem](#), this can possibly experience a hardware failure.
2. Use tmux or screen on the login and development VMs, especially when running tests.

## 2.7 How do I ask for assistance? Is there a mailing list?

Please refer to this [page](#) for more details on asking for help and posting to RG community groups and mailing lists.

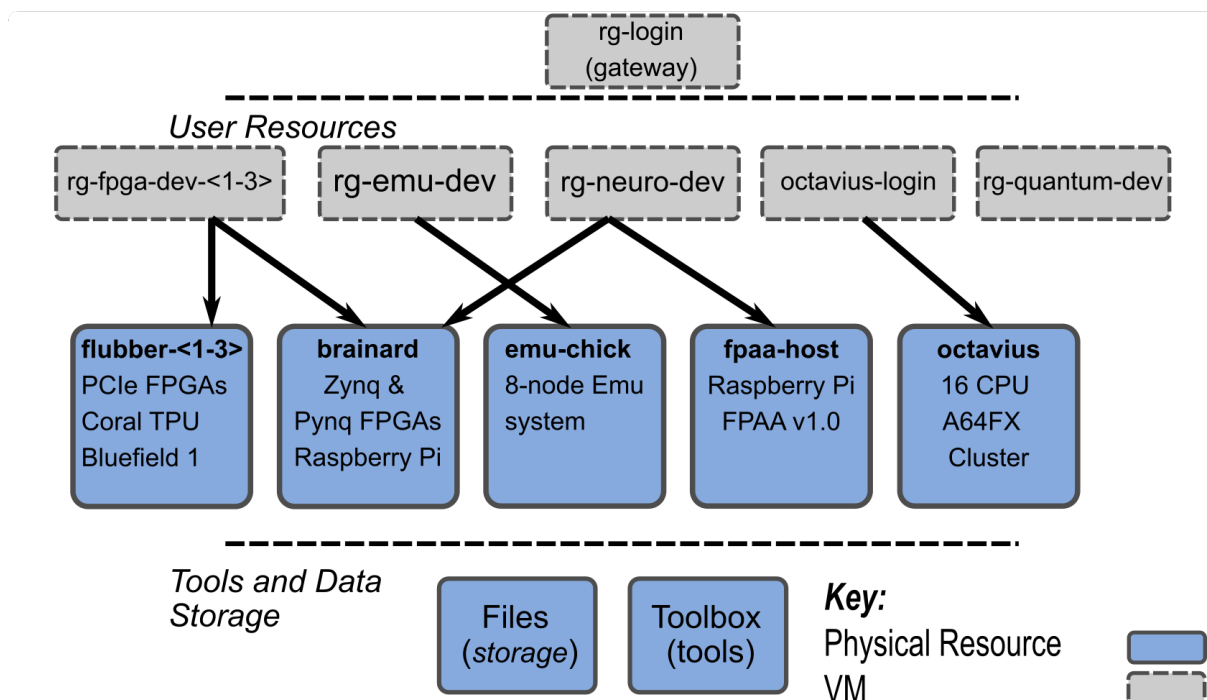




## ROGUES GALLERY HARDWARE

The Rogues Gallery testbed is composed of both “virtual” resources and “physical” servers and test platforms. The figure below shows the Rogues Gallery resources at a high level with suggested paths from development VMs to the actual hardware. Discussion of the filesystems you have access to can be found [here](#).

For instance, if you’re able to test your compilation and do debugging on the rg-fpga-dev VMs, you can then use the same code on the physical “flubber” boxes to test your code with a PCI Express-based FPGA. We currently have three pairs of development VMs that map to physical hardware, as described more in the tables below.



## 3.1 RG Virtual Machines

Note that VMs are meant as development resources since we may need to reboot physical hardware often to apply new drivers or reconfigure for specific experiments. These VMs have limited CPU cores and memory, but we can reconfigure them as needed or provide additional resources for long-running simulations. Local storage refers to space hosted within the VM for /localscratch. Except where noted, all machines are running RHEL8 or Ubuntu 20.04.

### 3.1.1 RG Virtual Machines

Resource	CPU	Memory (GB)	Local (GB)	Storage	Software and Features
rg-emu-dev	4 core, E312xx	4	33		Emu simulator and tools
rg-fpga-dev<1-3>	4 core, E312xx	16	250		RHEL 8 VM for FPGA tools
rg-fpga-dev4	4 core, E312xx	16	250		Ubuntu 20.04 VM for FPGA tools
rg-neuro-dev	4 core, Broadwell	8	450		ROS, FPAA tools
rg-quantum-dev	4 core, Broadwell	8	130		XACC, Qiskit

## 3.2 RG Physical Machines

### 3.2.1 Arm HPC

Resource	CPU	Memory (GB)	Network	Cards	Notes
octavius<1-16>	A64FX	32 GB HBM2e	EDR IB		

### 3.2.2 Near-Memory

The Pathfinder system consists of 4 chassis while the Emu Chick is a one chassis system. We recommend that you use the newer Pathfinder system for your work.

Resource	Nodes	Memory (GB)	LCEs	Network	Notes
pathfinder (4 chassis)	32	2048	784	RapidIO 2.0, 10 GE	
karrawingi (Emu Chick, 1 chassis)	8	512	64	RapidIO, 1 GE	karrawingi-login

### 3.2.3 Neuromorphic/AI

Re-source	CPU	Memory (GB)	Network	Cards	Notes
quorra1	2x <a href="#">AMD EPYC 7502 (Rome)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	Bluefield-2 NIC, 10 GE	4x A30	
quorra2	2x <a href="#">AMD EPYC 7502 (Rome)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	Bluefield-2 NIC, 10 GE	4x A30, 1x A100	
rg-fpaa-host	Raspberry Pi	2 GB			Hosts FPAA via USB UART connection
rudi1					<a href="#">Jetson Xavier NX Developer Kit</a>

### 3.2.4 Reconfigurable and Novel Networking

Re-source	CPU	Memory (GB)	Network	Cards	Notes
flub-ber1	2x <a href="#">Intel E5-2630</a>	256 DDR4, 2133 MHz, 32 GB DIMMs	10 GE	2x Alveo U50, Coral TPU	NA
flub-ber2	2x <a href="#">Intel Gold 6226R</a>	384 DDR4, 2666 MHz, 32 GB DIMMs	10 GE	Arria 10 PAC, Bittware 520N	NA
flub-ber3	2x <a href="#">Intel Gold 6226R</a>	384 DDR4, 2666 MHz, 32 GB DIMMs	10 GE, EDR IB	Stratix 10 PAC, Bittware 520MX, Bittware 385A-SoC	NA
flub-ber4	2x <a href="#">AMD EPYC 7513 (Milan)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	NVIDIA ConnectX-6	NA	NA
flub-ber5	2x <a href="#">AMD EPYC 7513 (Milan)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	NVIDIA ConnectX-6	Xilinx Alveo U280	NA
flub-ber6	2x <a href="#">AMD EPYC 7513 (Milan)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	NVIDIA ConnectX-6, Innova-2 Flex, <a href="#">Bluefield-1</a> NIC, Bluefield-2 NIC, Intel N6000	NA	Smart Networking Node
flub-ber7	2x <a href="#">AMD EPYC 7513 (Milan)</a>	256 DDR4, 3200 MHz, 16 GB DIMMs	NVIDIA ConnectX-6, Innova-2 Flex, <a href="#">Bluefield-1</a> NIC, Bluefield-2 NIC, Intel N6000	NA	Smart Networking Node
flub-ber8	2x <a href="#">Intel Gold 6338</a>	512 DDR4, 3200 MHz, 32 GB DIMMs	NVIDIA ConnectX-6	NA	NA
flub-ber9	2x <a href="#">Intel Gold 6338</a>	512 DDR4, 3200 MHz, 32 GB DIMMs	NVIDIA ConnectX-6	NA	NA
flub-ber10	2x <a href="#">Intel Gold 6338</a>	512 DDR4, 3200 MHz, 32 GB DIMMs	NVIDIA ConnectX-6	NA	NA

### 3.2.5 Devboard Hosts

Re-source	CPU	Memory (GB)	Net-work	Cards	Notes
brainard	i5-10210U	32 GB DDR4, 2666 MHz, 16 GB DIMMs	1 GE		Connection to RISC-V board and FPGA, <a href="#">Intel NUC10i5FNK</a>

### 3.2.6 RISC-V

Resource	CPU	Memory (GB)	Net-work	Notes
johnny-rv5-1	4xU74 cores and 1xS7 core, 1.4 GHz	16 GB DDR4	1 GE	SiFive Unmatched mother-board

### 3.2.7 Simulation and Tutorial Machines

Resource	CPU	Memory (GB)	Network	Notes
hawksbill / notebook	4x <a href="#">Intel E7-4820</a>	1024 DDR3	1 GE	Used for Jupyter notebooks

## 3.3 Techfee Systems

Re-source	CPU	Memory (GB)	Network	Cards	Notes
frozone<14>	2x <a href="#">Ice Lake 8352Y</a>	256 DDR4, 3200GHz, 16 GB DIMMs	<a href="#">Omni-Path 100 GB, 100HFA16LS</a>	<a href="#">1.6TB P5800X SSD</a>	FY 2021 TechFee Acquisition

## ROGUES GALLERY FILESYSTEMS

*Note:* We strongly encourage you to do compilation (especially for FPGAs) using `/netscratch` and back up your project files to the backed up “nethome” or `/home/<username>` directory. Code should additionally be backed up to a Github repo - remember the policy of keeping two separate copies of all important code/data!

### 4.1 Quotas and Data Retention Policies

#### 4.1.1 Quotas

We have the following quotas for users, subject to future modifications:

- `/home` - 50 GB is provided to each user and is backed up using ZFS snapshots and RAID.
- `/project` - Quotas vary but are typically in the 200 GB range depending on how many users are using a shared project.
- `/netscratch` - no quota limits but please note the lack of backups and data retention policies below. You can find a symlink to netscratch at `<user_home>/USERSCRATCH`

#### 4.1.2 Data retention policies

- Backups are provided via a standard fileserver with RAID and ZFS snapshot backups. *However*, we do not guarantee data backups and we recommend you use Github or GT Github for code backups and [GT’s Box service](#) or [PACE-provided storage](#) for large data backups.
- Deactivated accounts will have their data kept for 180 days (~6 months) and it will then be archived and removed at the discretion of the testbed maintainers.
- Scratch data will be *purged* after 120 days without user modification from both `/localscratch` and `/netscratch`. Note that scratch explicitly is not backed up, and we cannot restore deleted files.

### 4.2 Storage on CRNCH Systems

#### 4.2.1 Shared home

Each user has **50 GB** of shared storage space mounted under `/nethome/<user>` that is backed up via our data server (RAID 10) and that is served up via NFS. This storage is relatively stable and is meant for users to keep files consistent across Rogues Gallery resources. **NOTE:** We still strongly encourage users to back up their code using a git repo, such as those provided for free via Georgia Tech’s [github.gatech.edu](https://github.gatech.edu).

### 4.2.2 Shared tools

The Rogues Gallery VMs and physical resources are served by a shared `/tools/` directory that contains common tools for each architecture and that is shared via NFS to all available login, development, and experimental nodes.

### 4.2.3 Shared project space

Individual users may also request joint project space by emailing the admins at [crnch-rg-help@cc.gatech.edu](mailto:crnch-rg-help@cc.gatech.edu). These are served under `/projects/` and include space for data sets `/projects/bigdata` and other lab-based project groups.

### 4.2.4 Net scratch space

Speedforce.crnch.gatech.edu provides 47 TB of RAID 5 NVMe-backed storage that is mounted at `/netscratch` on each VM and CRNCH node. Please create a directory for your username and keep your data within this file. Note the data retention policies listed above that is enforced by modification time and cron job scripts. Any unmodified data older than **120 days** is subject to deletion, and users with very large data requirements may be asked to help purchase project-specific storage.

### 4.2.5 Local scratch space

Note that physical resources like `flubber` typically have their own fast scratch space, usually mounted at `/localscratch`. These are local scratch storage locations that are **not backed up** and are meant for temporary output that is then placed under a project or `nethome` folder. VMs also typically will have a `/localscratch` that is just using extra HD space in the VM that is a bit faster than the backed up `/nethome` but that is slower than `/netscratch`.

We recommend that you use `/netscratch` instead of local scratch as this is shared across all machines. Both types of scratch have the same data retention policies.

## MAILING LISTS AND REQUESTING HELP

### 5.1 Help and Mailing List FAQs

- Request access to the MS Teams group. If you are external to GT, please send us a ticket to add you.
  - We also have a CRNCH RG Slack workspace [here](#) if you have trouble connecting to Teams. Most of the Lucata discussion currently uses this Slack. Just note that since we are on a free plan that old conversations are not saved.
- How to use the help ticket webform at <https://crnch-rg.cc.gatech.edu/crnch-rg-help/>
- The main CRNCH help ticket email address is [crnch-rg-help@cc.gatech.edu](mailto:crnch-rg-help@cc.gatech.edu). We would prefer if you use the webform but the email address is ok if you include your GT username (e.g., [gburdell3](#) for George Burdell).

### 5.2 Help for the CRNCH Rogues Gallery

The Rogues Gallery testbed is rapidly growing along with its support structures. In general, we try to respond to help tickets within 1 business day of your request. Note that the admins and researchers for this project work 9-6 ET Monday-Friday, so any requests outside of this timeframe will be delayed at least until the next business day. Tickets submitted over the weekend or on any GT holidays will be evaluated on the following business day. Please note that Georgia Tech is officially closed for the last week of each calendar year for the Christmas holiday.

### 5.3 How do I request help?

The Rogues Gallery uses a Request Tracker (RT) based ticketing system. You can submit tickets ideally via the [webform](#) or alternatively using the email address, [crnch-rg-help@cc.gatech.edu](mailto:crnch-rg-help@cc.gatech.edu). We would request that you submit a help ticket for anything that is not a discussion or minor clarification question. This helps us track specific issues for users and also is important as a metric for our funded testbed project.

That said, the general process for asking for help is:

1. Check through the appropriate documentation on the public documentation or the internal wiki to see if this answers your question.
2. Ask a clarification question in the related MS Teams group. For example, if you are working with Alveo boards you may want to post your question in the `fpga-xilinx` channel on Teams.
  - Alternatively, if you are not able to use Teams you can post your question on the related mailing list. So for Alveo boards, this would be [crnch-rg-reconfig@lists.gatech.edu](mailto:crnch-rg-reconfig@lists.gatech.edu)

3. During discussion on the Teams or email group we may move your question to the ticketing system or a private conversation if it is specific to your account. When possible, we'd like to keep most discussion in the appropriate channel as it provides a good reference for other users who may have the same issue.
  - Submit tickets to the [crnch-rg-help@cc.gatech.edu](mailto:crnch-rg-help@cc.gatech.edu) email address or ideally via the webform at <https://crnch.gatech.edu/crnch-rg-help>
4. As you get to more in-depth research, you may need to pose your question to an external vendor since most architectures in RG are very new and GT researchers and staff are constantly learning along with you. More information on specific vendor forums can be found from this [link](#).

## 5.4 How do I ask a good question?

This is not meant to be a put down! Asking your question in a clear and meaningful way is the easiest way to get assistance. We generally recommend the following for asking a clear suggestion:

1. Share the machine name you are having issues with: Ex: "Python notebooks don't run on rg-neuro-dev.crnch.gatech.edu"
2. If possible, share the output using quotes or an attached log file. **Please share output as text where possible instead of a screenshot!**. If we are trying to recreate a problem you have it is much easier to copy and paste text than decipher it from a screenshot.
3. Create a "minimum working/reproducible example" (MWE/MRE). In many cases, sharing a huge log file for a full application is not useful to you or to the maintainers of a particular system. See this page on creating an effective MWE: [MWE overview](#)
4. Don't just state "Feature X doesn't work!". We'll ask you more follow-up questions so it's best to share more detail initially to get your question answered in a prompt fashion.

## 5.5 Rogues Gallery Mailing Lists and Teams Group

There are mailing lists for general user notifications and updates as well as sub-lists for each large area of research within the Rogues Gallery testbed infrastructure.

Please note that the MS Teams group for [CRNCH Rogues Gallery](#) may be the quickest way to post questions. However, the GT "SYMPA" mailing lists can also be used for discussion and will be used for subarea-specific announcements.

## 5.6 Mailing lists

To subscribe/unsubscribe from a mailing list please use this [SYMPA FAQ page](#). Note that most lists are closed to external users.

- [crnch-rg-users@lists.gatech.edu](mailto:crnch-rg-users@lists.gatech.edu) - general user list. Mainly used for announcements and very high-level discussion.
- [crnch-rg-fpga@lists.gatech.edu](mailto:crnch-rg-fpga@lists.gatech.edu) - Discussion of FPGA and reconfigurable computing
- [crnch-rg-nearmem@lists.gatech.edu](mailto:crnch-rg-nearmem@lists.gatech.edu) - Discussion of the Emu system and near-memory computing hardware like HBM, HMC, and NVDIMM
- [crnch-rg-networking@lists.gatech.edu](mailto:crnch-rg-networking@lists.gatech.edu) - Discussion of in-network computing, FPGA networking, and edge computing/5G



- [crnch-rg-neuro-sysml@lists.gatech.edu](mailto:crnch-rg-neuro-sysml@lists.gatech.edu) - Discussion of neuromorphic computing and systems for machine learning
- [crnch-rg-quantum@lists.gatech.edu](mailto:crnch-rg-quantum@lists.gatech.edu) - Discussion of quantum computing

## 5.7 Vendor Forums

If we can't answer your question via chat, we may direct you to the relevant vendor forum for more assistance. See a list of related vendors [here](#).



## VENDOR FORUMS

In many cases, we hope that the Rogues Gallery Teams group can be helpful to solving basic issues with novel hardware. However, at some point it is likely that you will need to post on a vendor-related forum. We have collected links to specific forums here and encourage you to sign up for the architecture you are working on.

### 6.1 Intel FPGA

- Intel’s main forum can be found [here](#). Note that the <https://community.intel.com/> site replaces Altera’s old forum.
- Intel vLab clusters or Intel HARP has a [ReadTheDocs](#) and you can email them at [iam@intel-research.net](mailto:iam@intel-research.net) for questions specific to the cluster environment. HARP specifically has a private user forum [here](#).
- OPAE has some support resources listed on their [Github for the OPAE SDK](#), but the mailing list seems inactive.
- Intel’s general support resources site can be found [here](#).

### 6.2 OneAPI

- Intel has a [community forum for OneAPI](#) and also a Discord that they use for workshops and tutorials.

### 6.3 Xilinx FPGA

- [Xilinx forums](#)
- [Community portal](#) - Includes link to forums and open-source designs

### 6.4 RISC-V Forums

- [Chipyard Google Group](#) - for Chipyard-specific issues
- [RISC-V Hardware Development Google Group](#)
- [Chisel Google Groups](#) - for questions specific to Chisel usage and debugging
- [Chisel Gitter](#)
- [Chisel Community Resources](#)

## 6.5 Lucata Corporation

- Check out the CRNCH RG Slack!
- Lucata has a private issue tracker on Github, if you have bugs to report. Please ask about this in the Teams channel, and we will help you post your issue.

## 6.6 Neuromorphic Computing

- Nengo forum

## 6.7 Mellanox Networking

- Community forum

## 6.8 Arm

- Community forum
- Arm HPC User Group Slack

## CONTRIBUTING TO THIS DOCUMENTATION

We welcome and appreciate your comments and contributions to this documentation. Often, we rely on a mix of vendor support and community support for novel architectures and in many cases *you* are the expert just by starting to work with this new, novel architecture or software stack. This page presents some suggestions for contributing your own documentation.

### 7.1 What is the process for contributing documentation?

To contribute documentation, you should write an rst-formatted file related to your topic and submit it via [pull request to this repository](#). We will also eventually have a form where you can upload and submit new documentation to be posted in a simplified fashion.

All read-the-docs documentation uses [Restructured Text](#), so we ask that you write documentation in this format or convert Markdown files for your pull requests. However, we can also accept updates in Markdown format and integrate them into this repo.

#### 7.1.1 Current Topic Areas:

- Emu/Lucata
- Getting Started - general topics needed to get started with RG
- Miscellaneous
- Networking - in-network computing, 5G, and edge.
- Neuromorphic and Novel Systems for ML
- Reconfig - all FPGA-related documentation
- Quantum
- Related - all related work

### 7.1.2 How do I update existing documentation?

- 1) Fork this repo.
- 2) Find the page you are looking to update and make your edits.
- 3) Add the page to the index file.
- 4) Build the code to test your code locally (requires Python and sphinx).
- 5) Commit *and* push to your fork and initiate a pull request.

### 7.1.3 How do I contribute new documentation?

- 1) Fork this repo.
- 2) Find the folder you are looking to update and create a new page with the following titleformat: *[Topic Area]-my-new-documentation-page*.
  - \* As an example, creating a new page for reconfigurable resources might have a title like:  
*[Reconfig]-Using-Vitis-for-AI.rst*
  - \* Note that you may need to create your page with the appropriate .rst suffix name and then edit it to get appropriate formatting in the browser. You can check basic format of commits in the Github web interface.
- 3) Commit *and* push to your fork and initiate a pull request.

## 7.2 What should my documentation contain?

- 1) A useful title
  - \* Ex: “Compiling ROS for the AC701 FPGA platform”
- 2) A brief description of the problem you’re trying to solve.
  - \* Ex: “This documentation details how to compile ROS for AC701 using the flubber server”.
- 3) Code snippets that demonstrate proper execution of a technique or application for the topic in question. See the suggested format for code snippets below.
- 4) References to relative links from official documentaiton or examples, if they are relevant.

### 7.3 Where should I add new files?

Ideally, your file should go under a directory related to your given topic. Please name your file with hyphens if it is a long filename, e.g., *reconfig-new-quartus-tools.rst* and place it under the correct folder in your commit. If you’re not totally sure where a file should go, feel free to submit a pull request and we can help you format and find the right place for your file.

### 7.4 How to convert Markdown files to RST

Note that we prefer RST formatted documentation since that is used natively by ReadTheDocs. However, if you can only submit Markdown documentation this is also very helpful and appreciated!

While there are many nice GUI-based editors for Markdown files, there are not that many editors for Restructured Text. One approach to contributing documentation would be to write it in Markdown and then convert it with Pandoc. [This URL](#) shows a simple example that we have repeated below.

```
pandoc rg-overview.md --from markdown --to rst -s -o rg-overview.rst
```

*#You can also use the shortened version of this command and pandoc will infer the right syntax*

```
pandoc -s rg-overview.md -o rg-overview.rst
```

You can use this simple script to convert a Markdown file to RST, but you may then want to check that the links and figure links get populated correctly. You can also use editors like [Typora](#) to edit Markdown and then export an RST file. Note that Typora uses pandoc behind the scenes to do this conversion!

You can also use Python tools like [m2r](#) which may provide a better translation capability than pandoc.

## 7.5 Style Guide Suggestions

### 7.5.1 How do I add a figure?

- 1) Add your figure to the docs/figures directory with a reasonable title. As an example, the overview page uses the figure `RG_CCRI_Infrastructure_Overview.png`.
- 2) Use RST syntax to add it the appropriate page.

```
#Adds the image with a relative path to the figure directory.
#Alt tag is nice to have if the image does not load correctly
.. figure:: ../figures/general/RG_CCRI_Infrastructure_Overview.png
:alt: RG CCRI Infrastructure Overview
```

- 3) Commit your change and check the hosting page to see if it looks reasonable (placement, height, width, etc.). Proceed with the pull request as normal.

### 7.5.2 How do I add code snippets?

You can use either two `` to delineate small segments of RST code or the “`.. code::`” tag to add indented code blocks. Check the figure instructions above for an example!

## 7.6 Questions?

Please email us via our ticketing system [crnch-help@cc.gatech.edu](mailto:crnch-help@cc.gatech.edu) with any questions or please feel free to ask on our CRNCH Rogues Gallery MS Teams group.

### 7.6.1 What are all these errors from Sphinx about “Title underline too short”

Errors like *Title underline too short.* mean that the header underline needs to be the same length as the text. This is a strict requirement for Sphinx, which usually results in a warning.



## TESTBED RELEASE NOTES

This page provides a high-level update on hardware and software that has been added or updated.

### 8.1 September 2023

#### 8.1.1 Hardware Updates

- Rudi2, an AGX Orin Devkit, has been added to the testbed

### 8.2 August 2023

#### 8.2.1 Software Updates

- Open OnDemand 3.0.1 installed and deployed (upgrading from 2.0.32)

#### 8.2.2 Hardware Updates

#### 8.2.3 Documentation Updates

- VS Code instructions updated along with instructions for SSH jump host usage

### 8.3 July 2023

#### 8.3.1 Software Updates

- OneAPI 2023.1 Base and HPC Toolkits have been installed system-wide.
- New modulefiles added for Quartus FPGA tools available on relevant nodes.
- [New VM for CI workflows](#) has been added to the testbed.
- ROCm 5.6 installed on [instinct](#).
- Lucata Pathfinder Toolkit 27.03 has been installed for relevant nodes.
- Xilinx Vitis, Vivado, and Vivado HLS 2022.2 and 2023.1 toolchains installed and updated.

### 8.3.2 Hardware Updates

- [Violet nodes](#) have been added as a new research resource with Sapphire Rapids CPUs.
- Bittware 840F Agilex FPGA card has been installed in flubber8.

### 8.3.3 Documentation Updates

- [VTune profiling page](#) added
- [Container migration to PACE page](#) added

## 8.4 June 2023

### 8.4.1 Software Updates

- NVHPC 2023.5 SDK installed on all NVIDIA GPU nodes.

## RG DOCUMENTATION AUTHORS

### 9.1 Current maintainers:

- Jeff Young

### 9.2 Community Contributors (listed alphabetically):

- Yujen Juan, Lucata Corporation
- Simrat Kaur, Illinois Tech
- Jason Riedy, Lucata Corporation
- Ron Rahaman, Georgia Tech
- James Wood, Georgia Tech



## USING SLURM WITH RG SYSTEMS

See [this page](#) with examples of how to use Slurm to request jobs on the testbed!

### 10.1 What is Job Scheduling and Why Do We Use It?

Job scheduling helps us to manage a limited number of novel resources with an active userbase while guaranteeing the resources you need to finish your job. While job scheduling is currently most used for homogeneous cluster resources like PACE's [Phoenix cluster](#) we have focused on using it for the Rogues Gallery to provide fair access to all users and to help document how specific resources are utilized.

We use [Slurm](#) as our job scheduler and resource manager as it is widely used by large cluster installations including Cori and Perlmutter at NERSC, Frontera at TACC, and near-term systems like Frontier at ORNL. Since the Rogues Gallery has a large diversity of resources, we have many different *workflows* depending on the novel architecture that is targeted.

### 10.2 What Slurm queues are available?

You can check the current status of all queues by using `sinfo --federation` on any RG node. The Pathfinder runs as a “federated” [Slurm cluster](#) since it has a different host environment than the other RG nodes.

## 10.3 Slurm Partitions

Queue Partition	Time Limit	Node:	Node List	Notes
rg-pathfinder	No limit	32	c0n[0-7],c1n[0-7],c2n[0-7],c3n[0-7]	Lucata Pathfinder system
rg-arm-debug	4 hours	4	octavius[1-4]	Used to compile/profile code
rg-arm-long	12 hours	16	octavius[1-16]	Used for longer running jobs
rg-gpu	12 hours	7	frozone[1-4],instinct,quorra[1-2]	Different types of GPU-focused nodes for classes and neuromorphic research
rg-hpc	12 hours	3	flubber[8-10]	General HPC nodes
rg-intel-fpga-hw	12 hours	2	flubber[2-3]	Hosts Intel FPGAs
rg-xilinx-fpga-hw	12 hours	2	flubber[1-3]	Hosts Xilinx FPGAs
rg-smart-nic	12 hours	2	flubber[6-7]	SmartNIC nodes
notebook	12 hours	1	hawksbill	Used to run Jupyter notebooks for tutorials, data analysis

## 10.4 How do I get started with Slurm on RG?

We suggest that you first check out the following Slurm “Getting Started” resources from LLNL if you have not used a batch submission system before.

Then please check out our RG Slurm Examples page and the RG Workflows page for architecture of interest and specific commands to run for these systems.

- [RG Slurm Examples](#)
- [RG Workflows](#)

### 10.4.1 Important Slurm Commands

Please consider looking at [PACE’s training information](#) for Slurm as well.

- `sinfo` - See status of queues and what is active/idle.
- `squeue` - See the status of your jobs. You can also run `squeue -u <username>` to just list your jobs.
- `scancel` - Used with the JOBID reported by `squeue` to cancel a job.

Options to run jobs include the following commands: - `salloc` - request resources from the Slurm scheduler and run a task when resources are ready - `sbatch` - create a batch file for later execution of one or more programs - `srun` - run parallel tasks across multiple processes. Called after `salloc/sbatch`.

## Slurm General Resources

- [Slurm Quickstart User Guide](#)
- [LLNL Slurm User Manual](#)
- [LLNL Slurm QuickStart Guide](#)
- [LLNL Slurm Commands Reference](#)
- [University of Maryland Torque versus Moab Guide Reference](#)
- [Princeton Research Computing's Slurm learning resources](#)
- [Slurm Video Tutorials](#)





## OPEN ONDEMAND

With the arrival of Slurm scheduling for most resources, we have switched to using Open OnDemand (sometimes called OOD) for GUI and notebook interfaces. [Open On Demand](#) provides a web browser interface from which users can launch GUI-based applications including Jupyter notebooks, file browsers, command-line terminals, and VNC-based applications. We encourage you to review the [general PACE instructions](#) for more details on what you can do with OOD - this page covers RG-specific tasks that you might find useful.

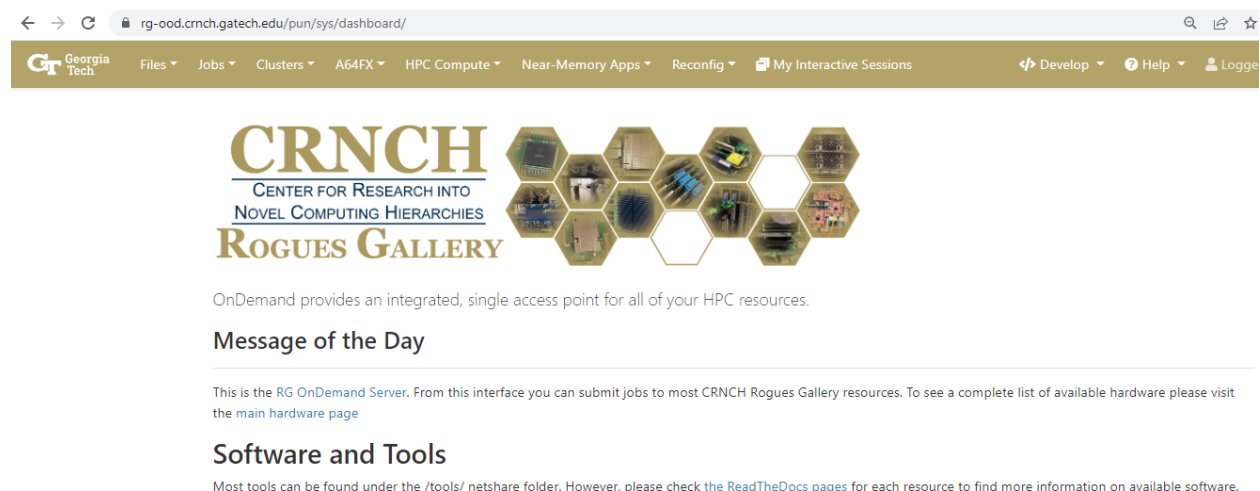
---

**Note:** OOD depends on using Slurm to reserve parts of nodes. This means that you will need to predict how long your job will take and how many cores your job needs. A compile that may take 3-4 hours should be run with a job that lasts at least that long.

---

### 11.1 Logging In

You should be able to log into the CRNCH RG Open OnDemand portal using your GT login and password with `rg-ood.crnch.gatech.edu`. If you are a GT employee or student, you will also need to use two-factor authentication (i.e., Duo) with your login while guests are not required to use two-factor login. Once you log in, you will be greeted by the login portal which has several tabs at the top of the webpage.



The screenshot shows a web browser window with the URL `rg-ood.crnch.gatech.edu/pun/sys/dashboard/`. The page features a navigation bar with the Georgia Tech logo and various menu items: Files, Jobs, Clusters, A64FX, HPC Compute, Near-Memory Apps, Reconfig, and My Interactive Sessions. On the right side of the bar are links for Develop, Help, and Logge. The main content area displays the CRNCH logo (Center for Research into Novel Computing Hierarchies) and the ROGUES GALLERY logo, which is a hexagonal grid of images showing various computing resources. Below the logos, a message states: "OnDemand provides an integrated, single access point for all of your HPC resources." This is followed by a "Message of the Day" section, which includes a link to the "RG OnDemand Server" and a link to the "main hardware page". The "Software and Tools" section follows, with a note that most tools can be found under the `/tools/ netshare` folder and a link to the "ReadTheDocs pages" for more information on available software.

## 11.2 VNC for GUI applications

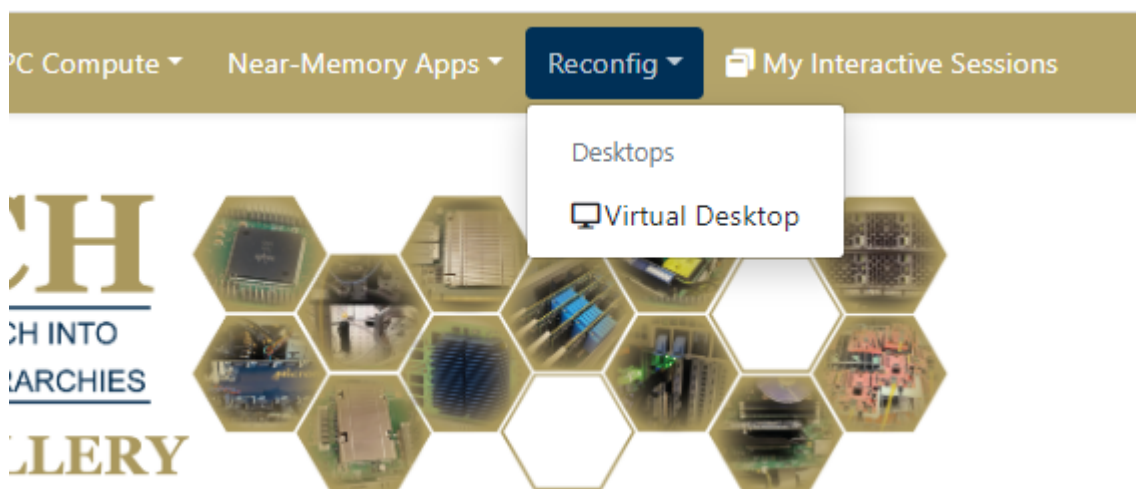
We have switched to using [VNC](#) instead of other tools like x2go because it is easier to schedule and works better with Open OnDemand.

---

**Note:** Like with x2go, your VNC session runs with a VNC server and will remain running for the length of your scheduled job, even if you need to close your laptop or the session tab. You can go to [My Interactive Sessions](#) to resume any running job.

---

To run a job on one of the reconfigurable server nodes, please select [Reconfig-->Virtual Desktop](#).



On the next page, you will need to specify the length of your job (we suggest 1-2 hours for non-compilation jobs), the number of cores to use (2-4 unless doing a parallel compile), and the queue (rg-xilinx-fpga-hw for flubber1 or rg-intel-fpga-hw for flubber2).

Once the job starts, you can change the image quality and compression settings for the VNC session to make your connection faster. Note that higher graphical quality may result in some lag on slower network connections. Click on [Launch Virtual Desktop](#) to start your VNC session.

When you are completely finished with your VNC job, you can go back to your [Interactive Sessions](#) tab and select “Delete” to kill the VNC server and your job.

## 11.3 Jupyter Notebooks

To use Jupyter notebooks on our notebook server, hawksbill, you can select [Near-memory Apps-->Jupyter Notebook](#).

You will then need to specify the time for your job (up to 12 hours) and the number of cores. We typically recommend starting with 2 cores unless you know you are running a parallel application from within your notebook.

Once the job launches, you can select [Connect to Jupyter](#) to connect to the Jupyter notebook interface.

You can then work within a Jupyter notebook to complete your project or experiments. Remember to save your notebook when you are finished and “Halt” when switching to other notebooks to save CPU resources.

## Virtual Desktop

This app will launch an interactive desktop on one or more Flubber compute nodes. You will have full access to the resources these nodes provide. This is analogous to an interactive batch job.

Account

Number of hours

Number of nodes

Queue

☐ I would like to receive an email when the session starts

Launch

\* The Virtual Desktop session data for this session can be accessed under the [data root directory](#).

Virtual Desktop (67110467)

1 node | 2 cores | Running

Host: >\_flubber1.crnch.gatech.edu

Created at: 2022-09-28 12:52:05 EDT

Time Remaining: 44 minutes

Session ID: ad921c6c-599f-44db-a96c-fd87169f1357

Compression  
0 (low) to 9 (high)

Image Quality  
0 (low) to 9 (high)

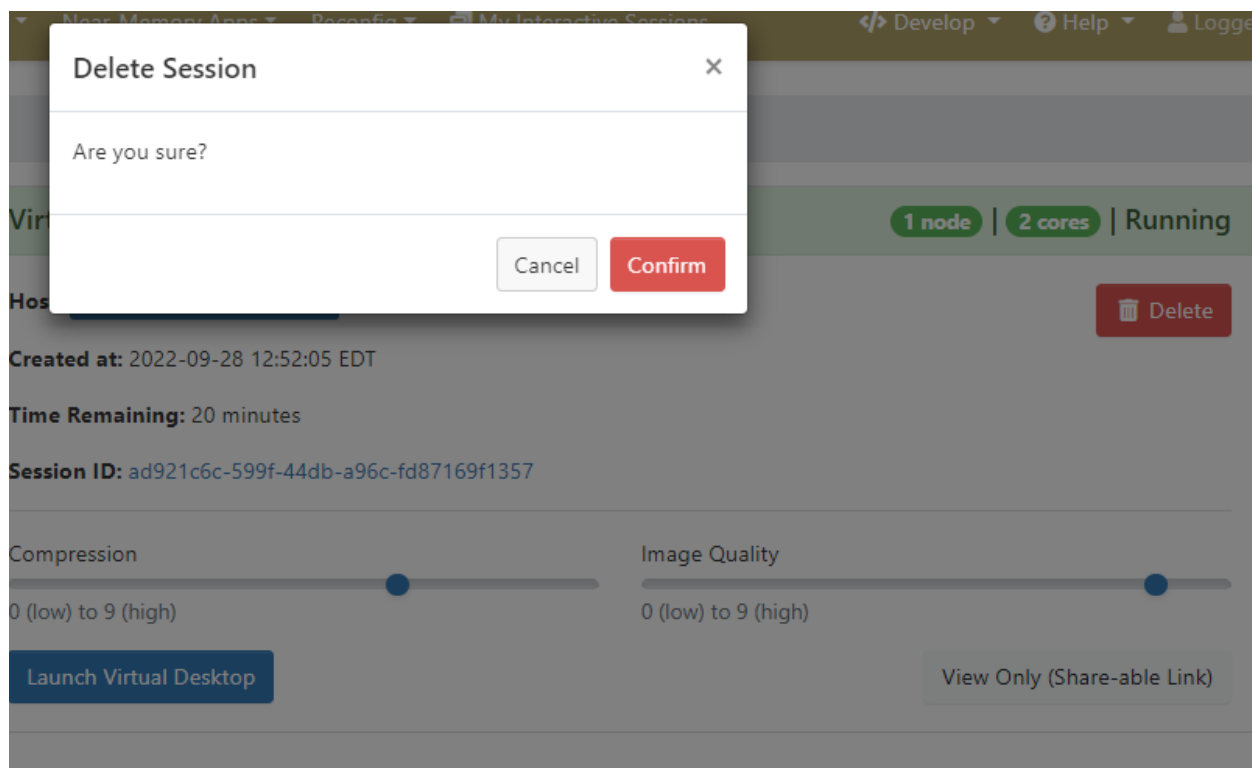
Launch Virtual Desktop

View Only (Share-able Link)

Delete

Fig. 1: You can then run applications from the terminal within the XFCE window or by using desktop icon launchers to launch GUI applications.





Home / My Interactive Sessions / Jupyter Notebook

## A64FX

Desktops

A64FX Desktop

GUIs

ARM Forge

Servers

Jupyter

Tutorials

A64FX Compiler Tutorial

## HPC Compute

Servers

## Jupyter Notebook

This app will launch a Jupyter Notebook server for emulation on the "notebook" partition.

Account

Number of hours

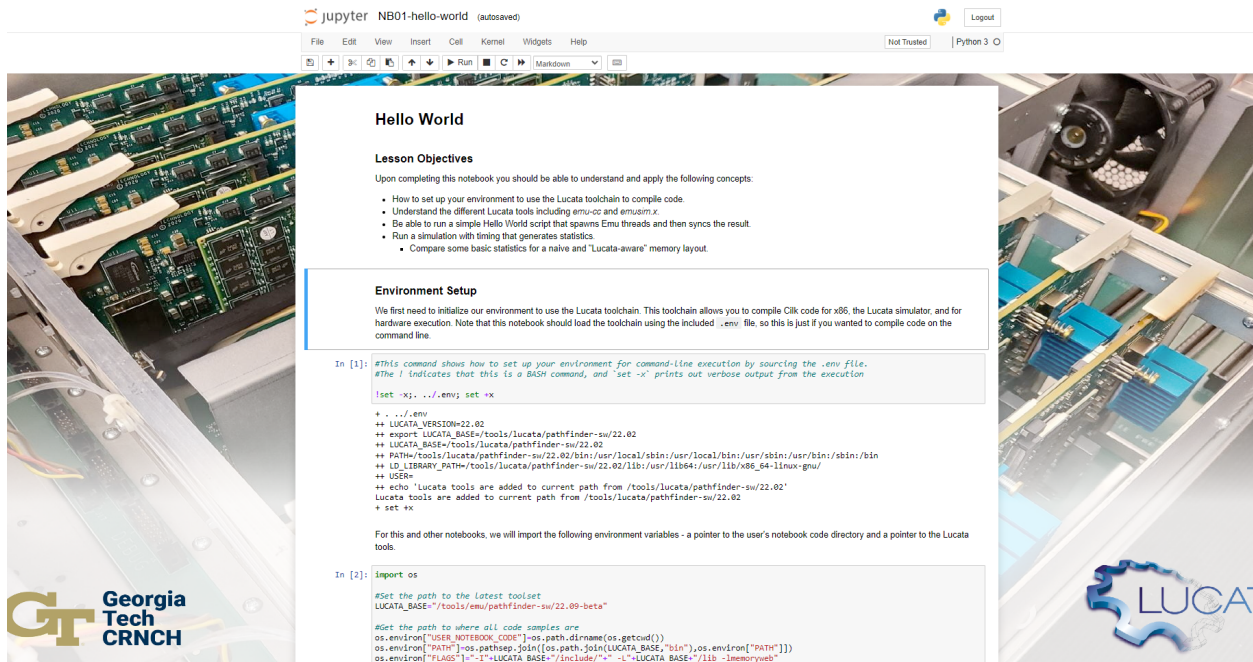
Number of vCPUs

Launch

\* The Jupyter Notebook session data for this session can be accessed under the [data root directory](#).

☐ 0
  / lucata-pathfinder-tutorial / code / 01-hello-world

- ☐ ..
- ☐ helpers
- ☐ NB01-hello-world.ipynb
- ☐ NB01.1-Plotting.ipynb
- ☐ hello-world-naive.c
- ☐ hello-world-spawn-at.c
- ☐ hello-world-spawn.c
- ☐ hello-world.c



**Hello World**

**Lesson Objectives**

Upon completing this notebook you should be able to understand and apply the following concepts:

- How to set up your environment to use the Lucata toolchain to compile code.
- Understand the different Lucata tools including `emu-os` and `emuasm.x`.
- Be able to run a simple Hello World script that spawns Emu threads and then syncs the result.
- Run a simulation with timing that generates statistics.
  - Compare some basic statistics for a naive and "Lucata-aware" memory layout.

**Environment Setup**

We first need to initialize our environment to use the Lucata toolchain. This toolchain allows you to compile Cilk code for x86, the Lucata simulator, and for hardware execution. Note that this notebook should load the toolchain using the included `.env` file, so this is just if you wanted to compile code on the command line.

```
In [1]: #This command shows how to set up your environment for command-line execution by sourcing the .env file.
#The ! indicates that this is a Bash command, and 'set -x' prints out verbose output from the execution
!set -x; . ../.env; set -x

+ . ../.env
++ LUCATA_VERSION=22.02
++ export LUCATA_BASE=/tools/lucata/pathfinder-sw/22.02
++ LUCATA_BASE=/tools/lucata/pathfinder-sw/22.02
++ PATH=/tools/lucata/pathfinder-sw/22.02/bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
++ LD_LIBRARY_PATH=/tools/lucata/pathfinder-sw/22.02/lib:/usr/lib64:/usr/lib/x86_64-linux-gnu/
++ USER=
++ echo 'Lucata tools are added to current path from /tools/lucata/pathfinder-sw/22.02'
Lucata tools are added to current path from /tools/lucata/pathfinder-sw/22.02
+ set -x

For this and other notebooks, we will import the following environment variables - a pointer to the user's notebook code directory and a pointer to the Lucata tools

In [2]: import os

#Set the path to the latest toolset
LUCATA_BASE="/tools/emu/pathfinder-sw/22.09-beta"

#Get the path to where all code samples are
os.environ["USER_NOTEBOOK_CODE"]=os.path.dirname(os.getcwd())
os.environ["PATH"]=os.pathsep.join([os.path.join(LUCATA_BASE, "bin"), os.environ["PATH"]])
os.environ["FLAGS"]="--I="+LUCATA_BASE+"/include/"+"-L="+LUCATA_BASE+"/lib -lmemoryeb"
```

You can also open a terminal from the Jupyter notebook environment, which is useful if you are mixing C or other code in your notebook. The [Lucata Pathfinder Tutorial](#) has several examples of C-based code that can be run from the notebook or on the command line.



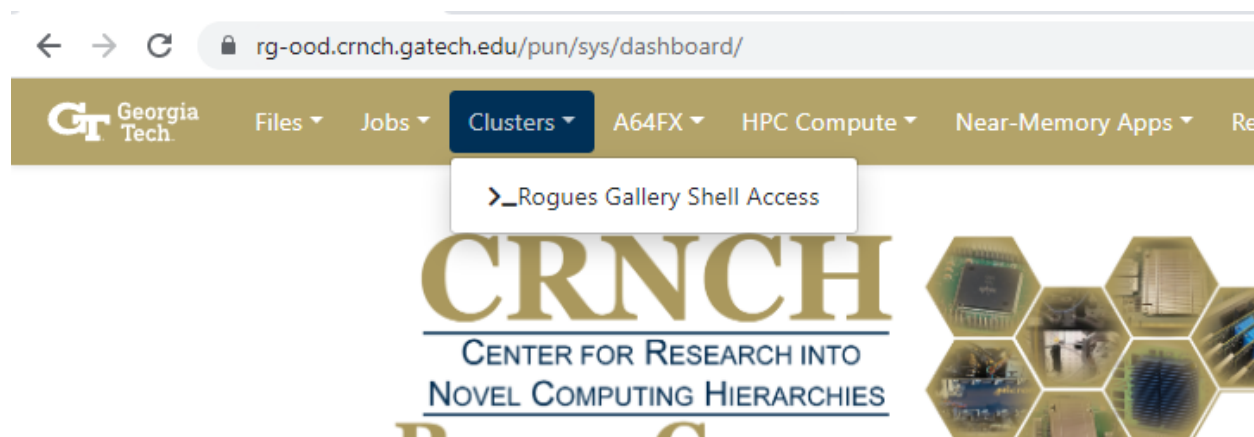
The screenshot shows the JupyterLab interface with the 'Clusters' tab selected. Below the tabs, there is a list of items to perform actions on: 'lucata-pathfinder-tutorial', 'ondemand', and 'CRNCH\_README.md'. A context menu is open over the 'CRNCH\_README.md' item, showing options: 'Notebook: Python 3', 'Other: Text File', 'Folder', and 'Terminal'.

## 11.4 Terminal from the Browser

To open a terminal on the main gateway node, `rg-login.crnch.gatech.edu`, you can select Clusters-->Rogues Gallery Shell Access.



```
crunch30@hawksbill:~$ /netscratch/tutorials/hpec-tutorial-2022/
lucata-pathfinder-tutorial/ update_pf_tut_repo.sh
crunch30@hawksbill:~$ /netscratch/tutorials/hpec-tutorial-2022/update_pf_tut_repo.sh
Updating a local copy of the Pathfinder tutorial repo
crunch30@hawksbill:~$
```



## 11.5 File Browser

Open OnDemand includes a file browser, that you can use to open and investigate certain files in your netshare folder and your netscratch folder.

## 11.6 Troubleshooting

- 1) **When launching a new job, the job may fail to launch with an I/O error.**
  - Failed to submit session with the following error: sbatch: error: Batch job submission failed: I/O error writing script/environment to file
  - Try to relaunch the job. This may just be related to small OOD bugs.
- 2) **When opening a VNC or Virtual Desktop session, you get the error “Failed to establish a websocket connection”.**
  - Clear the cache in your web browser, relogin and try to launch the job again. Specifically you may need to clear your cookies for gatech.edu domains.
- 3) If you are having trouble copying and pasting text from your local machine to the VNC window, please try opening the VNC session using Chrome. Firefox security policies seems to limit copy-paste to webpages using JavaScript.






[Open in Terminal](#) [+ New File](#) [New Directory](#) [Upload](#) [Download](#) [Copy/Move](#) [Delete](#)

[↑](#) / [nethome](#) / [crunch3](#) / [Change directory](#) [Copy path](#)

☐ Show Owner/Mode ☐ Show Dotfiles Filter:

Showing 3 of 16 rows - 0 rows selected

Type	Name	Size	Modified at
<input type="checkbox"/> 	<a href="#">lucata-pathfinder-tutorial</a>	-	9/22/2022 12:44:51 PM
<input type="checkbox"/> 	<a href="#">ondemand</a>	-	9/20/2022 4:37:09 PM
<input type="checkbox"/> 	<a href="#">CRNCH_README.md</a>	1.08 KB	9/14/2022 2:00:04 PM

### 11.6.1 More Resources

- [Using OOD for the Lucata Pathfinder tutorial](#)
- [PACE OOD Guide](#) - PACE's Open OnDemand instructions
- [Open OnDemand Discourse](#) - community discussions and a good place for Q&A



## VISUAL STUDIO CODE

---

**Note:** For remote GUI applications, we currently recommend using [Open OnDemand](#).

---

You are welcome to use [Visual Studio Code \(VS Code or VSC\)](#) along with the [Remote Development Extension Pack](#) to SSH to Rogues Gallery resources. This allows you to navigate folders and edit code with the power and convenience of VS Code and its many excellent extensions.

**Warning:** We highly recommend that you add a jump host to *hawkskill* and run VS Code from that node rather than on the login node, *rg-login*. Please see [this page](#) about setting up SSH jump hosts for this purpose.

Recommended Extensions Include:

- Remote Development - Includes the Remote SSH, Remote Container, and Remote WSL extensions
- Remote SSH - required to interact with remote SSH sessions on RG
- Jupyter - can be used to interact with and execute Jupyter notebooks from VSC.

Optional Extensions May Include:

- C/C++ - C syntax highlighting
- Python - Python highlighting

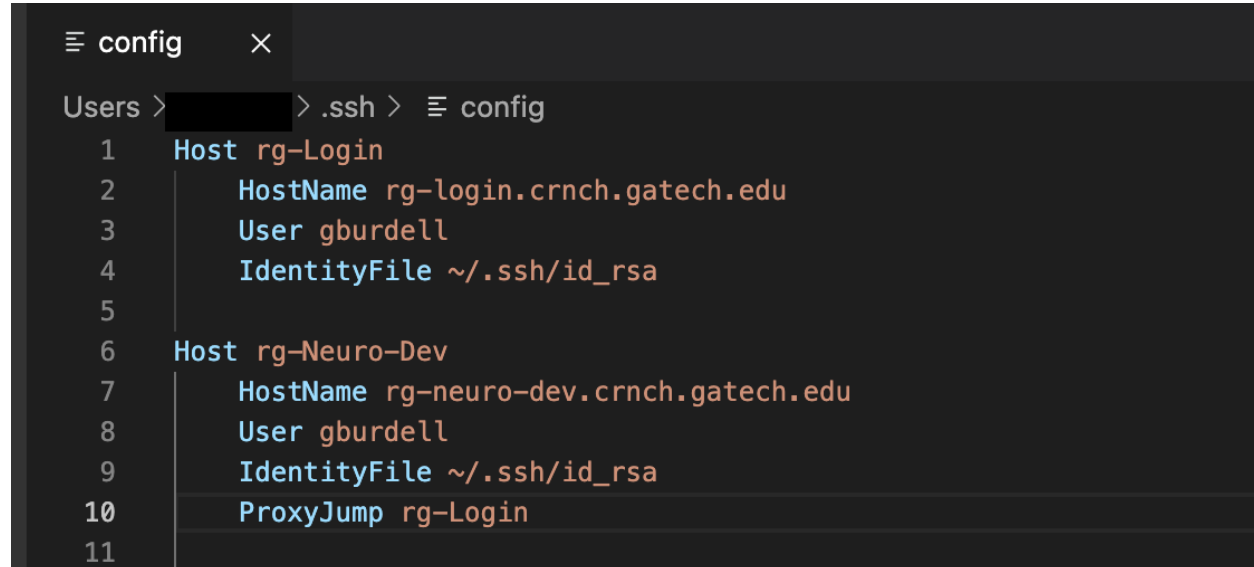
### 12.1 VSC and SSH Config Files

If you use VS Code, it is recommended to set up an SSH configuration file for easy access to RG nodes. In this way, you don't have to remember hostnames or worry about whether you are on the campus VPN or not.

1. After installing the Remote Development Extension Pack, click the green button on the bottom left corner of the screen.
2. A dialog will open at the top, select **Connect** to **Host** then select **Configure SSH Hosts...**
3. You should have at least one **config** file listed here, if there are multiple then just select the one associated with your current user.
4. This will open the **config** file in the VS code editor, from here follow this format to add a host to your file:

```
Host rg-Login
HostName rg-login.crnch.gatech.edu
User gburdell
```

```
Host rg-Neuro-Dev
HostName rg-neuro-dev.crnch.gatech.edu
User gburdell
ProxyJump rg-Login
```



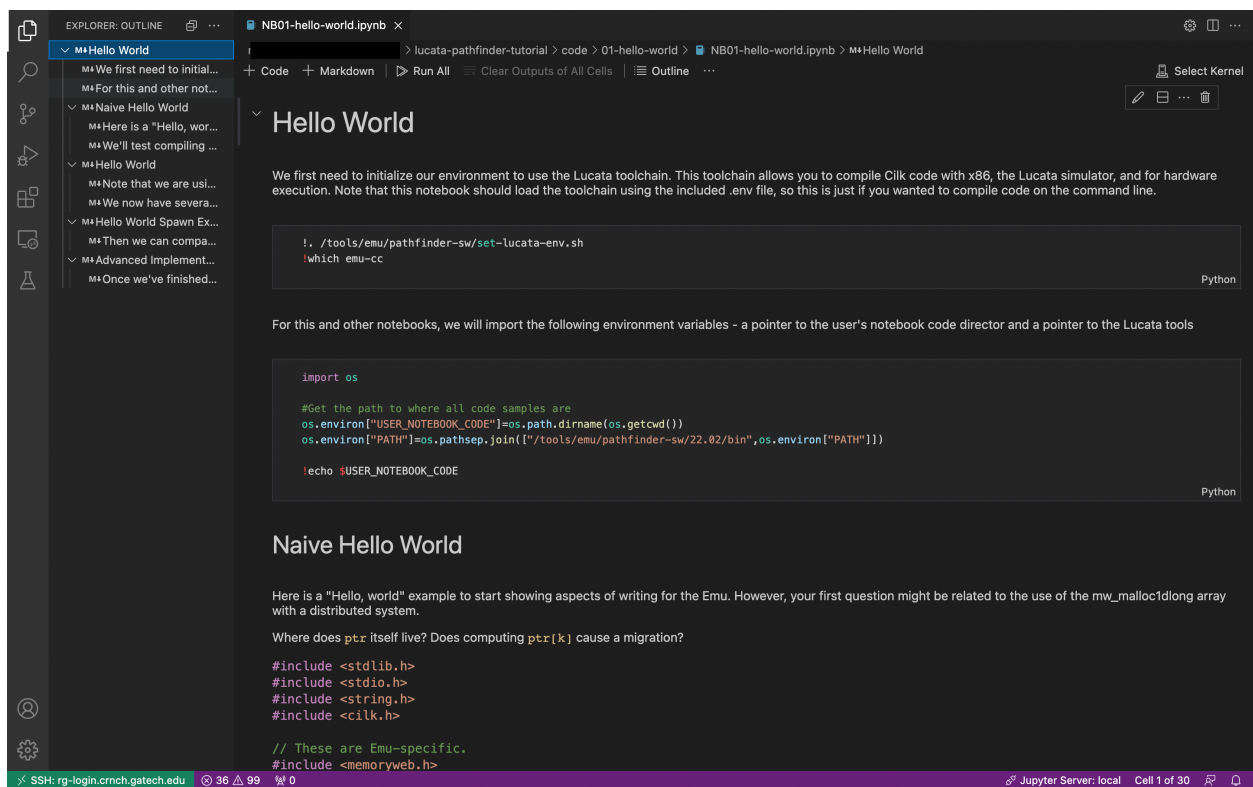
```
≡ config  ×
Users > > .ssh > ≡ config
1  Host rg-Login
2      HostName rg-login.crnch.gatech.edu
3      User gburdell
4      IdentityFile ~/.ssh/id_rsa
5
6  Host rg-Neuro-Dev
7      HostName rg-neuro-dev.crnch.gatech.edu
8      User gburdell
9      IdentityFile ~/.ssh/id_rsa
10     ProxyJump rg-Login
11
```

This has added `rg-login`, and below that `rg-neuro-dev`. Pay special notice to the `ProxyJump` option which will route your connection to `rg-neuro-dev` through `rg-login` first.

Now, if you select that green button again, then `Connect` to `Host`, you can pick either `rg-login` or `rg-neuro-dev` as options.

## 12.2 VSC with Jupyter Notebooks

*Special Thanks to our Contributing Authors for this Page:* James Wood





## PYTHON ENVIRONMENTS

---

**Note:** TLDR: we recommend that you use pipenv on CRNCH environments rather other Python package managers or tools like conda/anaconda. See the pipenv section for more details on usage.

---

### 13.1 FAQs on CRNCH Python Environments

- **The default Python is typically 3.8. We do not officially support Python 2 usage as most packages have updated to support Python 3.**
  - Look into [2to3](#) if your code is still using Python 2!
- **virtualenv and venv is installed across most of our servers and as many dev boards as possible.**
  - We recommend to use either virtualenv or venv with pip or pipenv to install packages into your local virtual environments. Note that venv and pip are default packages for all Python 3.3+ installations, and venv contains a subset of virtualenv functionality.
- **We do not typically recommend using conda, miniconda, or anaconda as these quickly eat up home directory space.**
  - However, if you want to use conda or miniconda please consider [using miniconda with your scratch space folder](#) to store your conda environment and venvs. See the example below under the Conda section as a template.

## 13.2 What's the difference between pip, venv, env, conda, etc?

Table 1: Python Environment Tools

Tool Name	Supported Python Versions	Purpose	Default on CRNCH RG	Notes
pip	All Versions	Default package manager	Y	
pipenv	2+	Package, dependency, and environment manager	Y	Combines pip and virtualenv
virtualenv	2+	Environment manager	Y	
venv	3.3+	Environment manager	Y	venv is a subset of virtualenv installed by default with Python 3.3+
mini-conda	NA	Minimalist package and environment manager	N	Suggested version of conda to use on RG; Installs its own conda/Python as well as non-Python packages
ana-conda	NA	Package and environment manager	N	Not supported on RG; Installs its own Python
poetry	3.7+	Package and dependency manager	N	Not supported on RG

## 13.3 Using venv on CRNCH RG

Venv is the default virtual environment module included since Python 3.3, and it totally replaces *pyenv* since Python 3.6. Virtualenv has many similarities to venv in terms of its functionality, but we recommend using venv unless you need to use a version of Python older than 3.3.

### 13.3.1 Creating a new virtual environment with venv

```
$ mkdir myproject
$ python -m venv myproject
# OR to create a specific Python version venv use the following command instead
$ mkdir my311project
$ python3.11 -m venv my311project
```

### 13.3.2 Activating/deactivating an environment

```
$> source myproject/bin/activate
//To leave type exit
(myproject)gburdell@rg-login:$ exit
//For our 3.11 version venv above
$> source my311project/bin/activate
(my311project) gburdell@rg-login:~$ python -V
Python 3.11.2
```



### 13.3.3 Installing and using packages

Here we demonstrate a basic usage of pip with venv. We highly recommend using pipenv, which provides a more robust combination of pip and virtual environments.

```
$ source myproject/bin/activate
(myproject)gburdell@rg-login:$ pip install matplotlib
Collecting matplotlib
Downloading matplotlib-3.6.2-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (9.
  ↳ 4 MB)
 9.4/9.4 MB 56.5 MB/s eta 0:00:00

//Use pip freeze to generate a requirements.txt file which can be used to reinstall a
  ↳ specific environment in the future.
pip freeze > requirements.txt
(myproject)gburdell@rg-login:~/USERSCRATCH/myproject$ ls
bin include lib lib64 pyenv.cfg requirements.txt share
(myproject)gburdell@rg-login:~/USERSCRATCH/myproject$ more requirements.txt
contourpy==1.0.6
...
matplotlib==3.6.2
numpy==1.24.1
...
six==1.16.0
```

## 13.4 Using pipenv on CRNCH RG

Pipenv combines the best parts of the pip package manager for Python and virtual environments, as typified by virtualenv and venv. One key difference is that pipenv keeps all of its dependencies for installations in a Pipfile that can then be used to regenerate a specific environment. Pipenv uses TOML syntax, and one Pipfile can be used in place of multiple requirements.txt files created by Pip with virtual environments. The Pipfile.lock file provides a secure hashed record of installations that can be used for future deployments.

### 13.4.1 Installing pipenv

Using the official installation instructions [here](#):

```
python3 -m pip install pipenv
```

### 13.4.2 Creating a new virtual environment with pipenv

When you run `pipenv install`, it will create a standard virtual environment and all related pip installs will occur within this user-accessible folder.

```
$ pipenv install
Creating a virtualenv for this project...
Pipfile: /nethome/gburdell/Pipfile
Using /usr/bin/python3.8 (3.8.13) to create virtualenv...
Creating virtual environment...created virtual environment CPython3.8.13.final.0-64.
```

(continues on next page)

(continued from previous page)

```

↳ in 2991ms
    creator CPython3Posix(dest=/nethome/gburdell/.local/share/virtualenvs/gburdell-
↳ hxKrwMjp, clear=False, no_vcs_ignore=False, global=False)
    seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle,
↳ via=copy, app_data_dir=/nethome/gburdell/.local/share/virtualenv)
        added seed packages: pip==22.3, setuptools==65.5.0, wheel==0.37.1
        activators BashActivator,CShellActivator,FishActivator,NushellActivator,
↳ PowerShellActivator,PythonActivator

✓ Successfully created virtual environment!
Virtualenv location: /nethome/gburdell/.local/share/virtualenvs/gburdell-hxKrwMjp
Pipfile.lock not found, creating...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)...
  0/0 - 00:00:00
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.

If you'd like to create a Python 3.8 environment, use the following syntax. Note that
↳ this will overwrite the standard location for your virtualenv

```

```
pipenv install pipenv --python 3.8 install
```

### 13.4.3 Activating/deactivating an environment

```

$ pipenv shell
Launching subshell in virtual environment...
. /nethome/gburdell/.local/share/virtualenvs/gburdell-hxKrwMjp/bin/activate
gburdell@rg-login:~$ . /nethome/gburdell/.local/share/virtualenvs/gburdell-hxKrwMjp/bin/
↳ activate
(gburdell) gburdell@rg-login:~$

```

OR use the code::*pipenv run* method

```

$ python3 --version
Python 3.6.8
$ pipenv run python3 --version
Python 3.8.13

```

### 13.4.4 Installing and using packages

```
$ pipenv install 2to3
Installing 2to3...
Adding 2to3 to Pipfile's [packages]...
✓ Installation Succeeded
Pipfile.lock (db4242) out of date, updating to (7d7dfd)...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
Building requirements...
Resolving dependencies...
✓ Success!
Updated Pipfile.lock (7d7dfd)!
Installing dependencies from Pipfile.lock (7d7dfd)...
  0/0 - 00:00:00
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

To show what packages are installed and their dependencies, you can use `pipenv graph`. Here we show the dependencies for `2to3` and `matplotlib`.

```
$ pipenv graph
//No dependencies for this package
2to3==1.0
//Several dependencies were installed, including numpy
matplotlib==3.6.2
- contourpy [required: >=1.0.1, installed: 1.0.6]
- numpy [required: >=1.16, installed: 1.24.1]
- cycycler [required: >=0.10, installed: 0.11.0]
- fonttools [required: >=4.22.0, installed: 4.38.0]
- kiwisolver [required: >=1.0.1, installed: 1.4.4]
- numpy [required: >=1.19, installed: 1.24.1]
- packaging [required: >=20.0, installed: 23.0]
- pillow [required: >=6.2.0, installed: 9.4.0]
- pyparsing [required: >=2.2.1, installed: 3.0.9]
- python-dateutil [required: >=2.7, installed: 2.8.2]
- six [required: >=1.5, installed: 1.16.0]
```

## 13.5 Pipenv Related Documents

- [Pipenv and Virtualenv](#)
- [Pipenv guide](#)
- [Pipenv vs virtualenv vs conda environment](#)

## 13.6 Pip

Pip or the *package installer for python* is the default way to install packages from the Python Package Index, or [PyPI](#). Depending on the version of Python used, you may need to call it using code::`pip install <packagename>` or code::`python -m pip install <packagename>`.

Note that best practices specify that you should install packages into a “user-local” directory (normally under `~/ .local` or your virtual environment folder). You can find this location for your version of Python using the following command.

```
$ python3 -m site --user-base
/nethome/gburdell/.local
```

Then you can install packages to your local directory as follows. Assuming a standard Python 3.8 install, the installed files can be found at code::`local/lib/python3.8/site-packages/`.

```
$ pip install --user matplotlib
```

### 13.6.1 Pip Related Documents

- [Python Pip tutorial page](#).

## 13.7 Conda

**Note:** We typically don’t recommend using anaconda due to the amount of dependencies it pulls into your home directory. If you get to where you need anaconda for a project this is typically some software that should be installed in a project space or system-wide! Please consider submitting a help ticket especially if you need multiple packages that can’t be satisfied with pipenv.

### 13.7.1 Miniconda Installation and Usage Example

With the above caveat in mind, this example shows how to use your scratch space to install and use Miniconda. We recommend this approach since this saves space in your home directory and because full Conda environments do not typically need to be backed up. Note that you can always use `conda env export --from-history>ENV.yml` to back up an installed environment.

```
mkdir ~/USERSCRATCH/conda
gburdell@rg-login:~/tutorials$ cd ~/USERSCRATCH/conda/
gburdell@rg-login:~/USERSCRATCH/conda$ wget https://repo.anaconda.com/miniconda/
↳ Miniconda3-py38_22.11.1-1-Linux-x86_64.sh
...
... 'Miniconda3-py38_22.11.1-1-Linux-x86_64.sh' saved [64630241/64630241]
//This command uses "batch mode" to auto-accept the EULA and installs in a local folder
gburdell@rg-login:~/USERSCRATCH/conda$ bash Miniconda3-py38_22.11.1-1-Linux-x86_64.sh -b_
↳ -p conda3_22.11.1
PREFIX=/nethome/gburdell/USERSCRATCH/conda/conda3_22.11.1
Unpacking payload ...
Installing base environment...
Downloading and Extracting Packages
...
installation finished.
```

(continues on next page)

(continued from previous page)

```
//Add the location of miniconda to your path. You should add this to your .bashrc file
export PATH=$PATH:~/USERSCRATCH/conda/conda3_22.11.1/bin && export LD_LIBRARY_PATH=$LD_
↳LIBRARY_PATH:~/USERSCRATCH/conda/conda3_22.11.1/lib

//Create a new conda environment on your scratch space.
conda create --prefix ~/USERSCRATCH/condaenv/
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##
environment location: /nethome/gburdell/USERSCRATCH/condaenv

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

#
# To activate this environment, use
#
#     $ conda activate /nethome/gburdell/USERSCRATCH/condaenv
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

### 13.7.2 Conda Related Documents

- Understanding conda and pip
- Explaining the many flavors of conda
- OLCF's guide to using conda, which assumes a sitewide installation of conda.
- NERSC's guide on using python with anaconda

## 13.8 Poetry

Poetry is a tool for dependency management and packaging similar to pipenv (which combines pip and venv). While we don't currently support it, you may be interested to try it out in your user-local setup. Read more about Poetry at the [official website](#).

## 13.9 Bonus: IPython, IPykernel, and Jupyter

You may see some reference to IPython kernels which switching between virtual environments or especially for Jupyter notebooks. In short, IPython ([see site](#)) is a command shell for interactive Python execution that can be extended for GUI applications and parallel computing. Jupyter is a web-based interactive tool that builds on IPython but also supports many other kernels for languages like Julia and R. You can read more about kernels for Jupyter [at this link](#).

## JUPYTER NOTEBOOKS

We suggest using JupyterLab as your interface for neuromorphic, quantum, and reconfigurable research. PACE has a [nice article](#) on using Jupyter notebooks we also recommend. Eventually we will support schedulable Jupyter notebooks via our Slurm scheduler.

*Note:* If you are running a long-running job like ML training we recommend that you don't use port forwarding as closing your local machine/web browser will likely stop or pause any actions within the Jupyter notebook. You alternatively can use [VNC](#) or [x2go](#) to create a remote server and you can then run your notebook within a browser in that remote terminal session.

### 14.1 Using jump hosts to access Jupyter notebooks

Note that these techniques will let you use “port forwarding” to forward the Jupyter notebook to your local machine.

First connect from your local terminal to this server using SSH forwarding for a specific port:

```
ssh -L 59801:localhost:59801 -C -J rg-login.crnch.gatech.edu rg-quantum-dev.crnch.gatech.edu
```

Then start your JupyterLab instance on the node you want to use a notebook with:

```
rg-quantum-dev:~$ jupyter-lab --no-browser --port 59801
[W 21:20:20.924 LabApp] JupyterLab server extension not enabled, manually loading...
[I 21:20:20.931 LabApp] JupyterLab extension loaded from /nethome/gtburdell/.local/lib/
python3.8/site-packages/jupyterlab
[I 21:20:20.931 LabApp] JupyterLab application directory is /nethome/gtburdell/.local/
share/jupyter/lab
[I 21:20:20.934 LabApp] Serving notebooks from local directory: /nethome/gtburdell/git_
repos/vip_class
[I 21:20:20.935 LabApp] Jupyter Notebook 6.1.5 is running at:
[I 21:20:20.935 LabApp] http://localhost:59801/?
token=f59c32ebcccbbc3f3036bfd32b8f62a47b48442085f3f4a2
[I 21:20:20.935 LabApp] or http://127.0.0.1:59801/?
token=f59c32ebcccbbc3f3036bfd32b8f62a47b48442085f3f4a2
[I 21:20:20.935 LabApp] Use Control-C to stop this server and shut down all kernels.
(twice to skip confirmation).
[C 21:20:21.002 LabApp]
```

To access the notebook, open this file in a browser:

file:///nethome/gtburdell/.local/share/jupyter/runtime/nbserver-14090-open.html

Or copy and paste one of these URLs:

(continues on next page)

(continued from previous page)

```
http://localhost:59801/?token=f59c32ebccccbc3f3036bfd32b8f62a47b48442085f3f4a2
or http://127.0.0.1:59801/?token=f59c32ebccccbc3f3036bfd32b8f62a47b48442085f3f4a2
```

Connect locally to the notebook using your web browser with the token provided.

```
http://localhost:59801/?token=f59c32ebccccbc3f3036bfd32b8f62a47b48442085f3f4a2
```

**Note:** Remember to close both your jump session and your jupyterlab session when you are finished!

## 14.2 Alternative method: use the provided setup script

This method provides a randomized port number for you to use and is preferred. It still requires creating a tunnel to your home machine but shares the string to do so with you:

```
./tools/misc/scripts/start_jupyter_notebook.sh
=====
Connect to this instance using port forwarding in another terminal from your machine.
→with the following string:
ssh -L 54050:localhost:54050 -C -J rg-login.crnch.gatech.edu notebook.crnch.gatech.edu
1) Then Connect your browser to http://localhost:54726/<yourtoken>
2) When done, use Ctrl-C and y to exit the jupyter port forwarding environment
=====

[I 14:35:34.137 LabApp] JupyterLab extension loaded from /nethome/gburdell/.local/lib/
→python3.8/site-packages/jupyterlab
[I 14:35:34.137 LabApp] JupyterLab application directory is /nethome/gburdell/.local/
→share/jupyter/lab
[I 14:35:34.140 LabApp] Serving notebooks from local directory: /net/tools/misc
[I 14:35:34.140 LabApp] Jupyter Notebook 6.1.5 is running at:
[I 14:35:34.140 LabApp] http://localhost:54050/?
→token=ccc0907e7f4f22ed7c25dbcbc63ce6512e34e3fbbca5c077
[I 14:35:34.140 LabApp] or http://127.0.0.1:54050/?
→token=ccc0907e7f4f22ed7c25dbcbc63ce6512e34e3fbbca5c077
[I 14:35:34.140 LabApp] Use Control-C to stop this server and shut down all kernels.
→(twice to skip confirmation).
[C 14:35:34.145 LabApp]

To access the notebook, open this file in a browser:
file:///nethome/gburdell/.local/share/jupyter/runtime/nbserver-1303810-open.html
Or copy and paste one of these URLs:
http://localhost:54050/?token=ccc0907e7f4f22ed7c25dbcbc63ce6512e34e3fbbca5c077
or http://127.0.0.1:54050/?token=ccc0907e7f4f22ed7c25dbcbc63ce6512e34e3fbbca5c077
```



## 14.3 Example images of Jupyter notebook usage

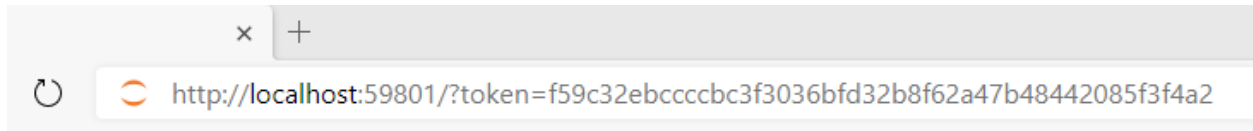


Figure 1: Example of using the JupyterLab token to connect locally

Figure 2: JupyterLab session with Qiskit code

JupyterLab interface showing a notebook titled "Random Circuit Gen.ipynb" running on localhost:59801/lab.

The notebook contains the following code cells:

```
[1]: import numpy as np
from qiskit import *
from qiskit.circuit.random import random_circuit
%matplotlib inline
```

```
[2]: qubits = np.random.randint(2,4)
depth = np.random.randint(1,10)
circ_U = random_circuit(qubits, depth)
circ = circ_U.combine(circ_U.inverse())
circ.measure_all()
```

```
[8]: circ_U.draw('mpl')
```

The output of cell [8] is a quantum circuit diagram for 3 qubits ( $q_0, q_1, q_2$ ):

- $q_0$  has a  $S^\dagger$  gate followed by a  $R_z$  gate with parameter 2.77.
- $q_1$  has a  $+$  gate.
- $q_2$  has a  $R_y$  gate with parameter 5.33.
- There are control points on  $q_1$  and  $q_2$  connected to the  $R_z$  gate on  $q_0$ .

```
[9]: circ.draw('mpl')
```

The output of cell [9] is a quantum circuit diagram for 2 qubits ( $q_0, q_1$ ):

- $q_0$  has a  $S^\dagger$  gate, followed by a  $R_z$  gate with parameter 2.77, then another  $R_z$  gate with parameter -2.77, and finally an  $S$  gate.
- $q_1$  has a  $+$  gate.
- There are control points on  $q_1$  connected to the first  $R_z$  gate on  $q_0$  and the  $S$  gate on  $q_0$ .
- Both qubits end with measurement gates.

## USING SCRONTAB WITH SLURM

### 15.1 What is Scrontab?

**Scrontab** is an extension of Linux's **cron** functionality that allows you to schedule Slurm jobs at regular intervals. For instance, maybe you want a slurm job to run and process data each day at a specific time - scrontab can help you set that up as follows in your *scrontab*:

```
0 11 * * * $HOMEDIR/myslurmtask.job
```

Note that the scrontab format follows that of crontab and cron. You can use sites like [crontab.guru](https://crontab.guru) to help define your scrontab rules.

### 15.2 ScronTab Todos

- note where scrontab has to live in user directory and any tmp directory suggestions
- any scrontab limitations
- how do we know if scrontab is enabled?
- design an example of using scrontab to pull data from a site like IBM's Q Experience.



## USING GUI APPLICATIONS WITH VNC

**NOTE:** Please also see our *page on using X2Go* as this is a little bit simpler to set up than VNC.

Here we cover how to use a VNC server on the CRNCH Rogues Gallery machines to access and manipulate GUI-based applications. We typically have *TightVNC servers* set up in conjunction with the *xfce4 desktop* as this allows for a low-latency but somewhat standardized and open-source setup.

### 16.1 Table of Contents

- *Quick Start*
- *What is VNC?*
  - *How does VNC differ from X11 forwarding?*
  - *How does VNC differ from NX?*
  - *How does VNC differ from Remote Desktop?*
- *VNC Clients*
- *Setting up a VNC server*
  - *Ubuntu*
  - *RHEL*
- *Setting up and using your VNC client (GT Network or VPN)*
  - *Using SSH jump hosts (VNC without the GT VPN)*
  - *Securing VNC Server*
  - *Killing a VNC server instance*
- *Using GUI jobs with the other GT Resources (PACE tips)*
- *Advanced tips*
- *Common Errors*

## 16.2 TLDR - How do I quickly get started using VNC on CRNCH resources?

This guide goes into great details on the how and why of using VNC. If you have questions, please refer to the following sections for more information. As a baseline just to get started you need to do the following:

1. Select and install a VNC viewing client on your local workstation. We recommend TightVNC for Windows or [RealVNC Viewer](#) for Mac OSX.
2. Connect to your destination server using SSH jumphosts and port forwarding. Here we use a jumphost to ssh to our destination server.

```
ssh -L 5901:localhost:5901 -C -J <yourGTusername>@rg-login.crnch.gatech.edu  
↪<yourGTusername>@flubber1.crnch.gatech  
.edu
```

1. Start the remote VNC server. On the first time you start this, you will need to enter a new password.

```
//On newer flubber installs you may need to add the newest VNC to your path. This will  
↪be updated shortly!  
export PATH=$PATH:/opt/TurboVNC/bin  
//Launch a vnc server  
vncserver  
#Enter your 6 character password on first use.
```

1. Using your local VNC viewer client connect to “localhost::5901” using your password. You then should be able to launch the web browser and apps.
2. Once you are done, please kill your VNC session in the terminal before logging out.

```
#Usually you will kill instance #1  
vncserver -kill :1  
#Check with ps aux to see if you have any VNC sessions open and kill these  
ps aux | grep <yourGTusername>  
kill <any pids for xtightvnc>
```

There is a [script](#), `crnch_client_vnc_login.sh` in the rogues-docs repository that you can use for the initial port forwarding setup from your local machine.

## 16.3 What about running GUI applications on Windows Subsystem for Linux?

This uses X forwarding and is described more in this [page](#).

## 16.4 What is VNC?

While there is a lot of good information about “virtual network computing (VNC)” [online](#), the main keys to remember are that VNC consists of a server running on the remote machine, a client running on your local host, and graphical updates and keystrokes are exchanged via a protocol called Remote Frame Buffer (RFB). There are many different optimizations for serving and updating a graphical environment using RFB with optimizations to only transfer data when a particular “square” of the served application changes.

## 16.5 How does VNC differ from X11 forwarding?

Using `ssh -X` or `ssh -Y` will allow you to use “X forwarding” to forward one application to your desktop. The main difference between X forwarding and VNC is that X forwarding typically only forwards one application at a time while VNC will forward a *session*, including the application and a desktop environment.

VNC sessions also typically survive a disconnection because they are running remotely on the VNC server rather than being forwarded to your local X server. This also means that VNC requires more resources on the server while X forwarding typically requires a faster client. VNC is also optimized to have lower latency which may make your session seem more responsive.

X11 sessions may handle multi-layered windows (think Matlab with an output figure) better than VNC because it will spawn multiple windows rather than just showing the graphical region that has been changed, as with VNC. This is a trade-off in terms of performance and usability.

For more differences please check out these links:

- A brief 2009 [post](#)
- Ars Technica 2010 [post](#)

## 16.6 How does VNC differ from NX?

NX is a similar protocol to RFB that is further optimized for low-bandwidth connections and that supports more options for secure connections (i.e. Kerberos, SSH keys). The NX protocol is proprietary and is created by the Belgian company NoMachine. However, free implementations of NX like FreeNX are available.

- [Wikipedia - NX Overview](#)
- [Ubuntu - What is FreeNX?](#)
- [Linux Journal 2005 article on NX](#)

## 16.7 How does VNC differ from Remote Desktop?

The main difference is that Remote Desktop is typically an OS-dependent tool that has slightly better integration with the video driver stack. Microsoft's remote desktop has been around since the early 2000s, and Apple's Remote Desktop is a more recent innovation. Both of these tools are typically used in an IT setting for "remote assistance".

- [MS Remote Desktop vs VNC](#)
- [Apple Remote Desktop vs VNC](#)

## 16.8 VNC Clients

A master list of remote desktop client comparisons can be found on [Wikipedia](#).

- TightVNC
- UltraVNC
- TigerVNC - Supports Mac
- NoMachine (NX)
- VNCViewer
- TurboVNC - Recommended by PACE

## 16.9 Setting up a VNC server

We suggest using xfce4 as the desktop environment because it is a bit more full-featured than standard X11 but still lightweight.

### 16.9.1 Ubuntu

Install the following:

```
sudo apt install tightvncserver xfce4 xfce4-terminal firefox autocutsel -y
```

### 16.9.2 RHEL

```
sudo dnf install tigervnc-server xfdesktop xfce4-terminal xfce4-session
```



## 16.10 Setting up and using your VNC client (GT Network or VPN)

This [guide](#) has a great introduction to setting up both a VNC server and your local client.

1. Set up your local VNC viewer client from the list above.
2. Connect to the remote server using the appropriate SSH port forwarding command. If you are connecting to a node *other* than rg-login or hawksbill, you will either need to be on the GT network (LAWN) or connected to the campus network with VPN.

```
# -L specifies which port to forward from the remote machine to the local machine
# -C specifies to use compression. This is usually useful for X sessions or VNC!
# Here we want to forward the GUI from rg-login to our local machine. We assume that the
# user is on GT network or connected to the VPN.
ssh -L 5901:localhost:5901 -C gtburdell@rg-login.crnch.gatech.edu
gtburdell@rg-login.crnch.gatech.edu's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-91-generic x86_64)
...
#Start the TightVNC server on the remote machine
[rg-login]$vncserver
#If it is the first time you've started this server, you will need to create a unique
# password for the server.
```

## 16.11 Using SSH jump hosts (VNC without the GT VPN)

Note that applications might be a little bit more responsive when you are connected to the GT VPN, but you can also use SSH jump hosts to connect directly to an application on a server behind the firewall.

```
# -L specifies which port to forward from the remote machine to the local machine
# -C specifies to use compression. This is usually useful for X sessions or VNC!
# -J <host1> <host2> specifies a jump host, where you log into host1 and then "jump" to
# host2
# Here we want to forward the GUI from flubber to our local machine.
ssh -L 5901:localhost:5901 -C -J gtburdell@rg-login.crnch.gatech.edu gtburdell@flubber.
# crnch.gatech.edu
gtburdell@rg-login.crnch.gatech.edu's password:
gtburdell@flubber.crnch.gatech.edu's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)...
...
#Start the TightVNC server on the remote machine
[flubber]$vncserver

New 'X' desktop is flubber:1

Starting applications specified in /nethome/gtburdell/.vnc/xstartup
Log file is /nethome/gtburdell/.vnc/flubber:1.log
#Now you can proceed to connect with your local VNC viewer using the address
# "localhost::5901"
```

## 16.12 Securing VNC Server

When you first set up a new VNC server instance, it should ask you for an 8 character password. Please note that this password is stored locally in your `~/vnc/passwd` file, and it is “encrypted” but not necessarily hard to reverse engineering. These limitations (password size, etc.) mainly ensure compatibility with the RFB protocol, so for this reason we strongly encourage that you kill `vncserver` sessions when you are done with them!

If you need to change your password, you can use the “`vncpasswd`” function to do so. Note that we advise not entering a “view-only” password.

```
$ vncpasswd
Using password file /nethome/jyoung9/.vnc/passwd
Password:
Verify:
Would you like to enter a view-only password (y/n)? n
```

## 16.13 Killing a VNC server instance

To kill the remote server you want to kill the particular “session” that is being served; usually this is the first session. However if you run multiple instances of `vncserver` you may have to use `:2`, `:3` etc. You can check to see which `vncservers` are running with `ps aux | grep vnc`.

```
#Checking to see how many VNC sessions are running
ps aux | grep vnc
gtburdell  2452  0.0  0.1  57752 15864 pts/0    S   13:04   0:00 Xtightvnc :1 -desktop_
↪X -auth /nethome/gtburdell/.Xauthority -geometry 1024x768 -depth 24 -rfbwait 120000 -
↪rfbauth /nethome/jyoung9/.vnc/passwd -rfbport 5901 -fp /usr/share/fonts/X11/misc/,/usr/
↪share/fonts/X11/Type1/,/usr/share/fonts/X11/75dpi/,/usr/share/fonts/X11/100dpi/ -co /
↪etc/X11/rgb
#Kill the first VNC session
vncserver -kill :1
Killing Xtightvnc process ID 1922
```

## 16.14 Using GUI jobs with the other GT Resources (PACE tips)

PACE has other resources for running graphical jobs including tips on X forwarding and submission scripts like `pace-vnc-job` and `pace-jupyter-notebook` which allow users to use VNC and SSH port forwarding to view graphical applications and Jupyter notebooks hosted on PACE cluster interactive jobs, respectively.

- [PACE’s Interactive Jobs with VNC](#)
- [PACE Port Forwarding Guide for Interactive Jobs](#)
- [PACE Jupyter Notebook Support](#)
- [PACE - X forwarding](#)

## 16.15 Advanced tips

- Using copy-paste between your local client and the remote VNC session: With TightVNC, this requires the install of the package `autocutsel` as detailed in this [post](#).
  - A sample `~/.vnc/xstartup` file that includes support for `autocutsel` is as follows:

```
#!/bin/bash
xrdb $HOME/.Xresources
#Use the xfce desktop instead of a basic X session, /etc/X11/Xsession
startxfce4 &
xrdb $HOME/.Xresources

# -solid grey gives us a real mouse pointer instead of the default X
xsetroot -solid grey -cursor_name left_ptr
# Allow copy & paste when ClientCutText is set to true on the client side
autocutsel -fork
```

## 16.16 Common Errors

```
startxfce4
/usr/bin/startxfce4: Starting X server

/usr/lib/xorg/Xorg.wrap: Only console users are allowed to run the X server
No protocol specified
xinit: giving up
xinit: unable to connect to X server: Resource temporarily unavailable
xinit: server error
```

`/etc/X11/Xwrapper.config` needs to be edited to allow “anybody” to run X11 instead of just console users.

TBD - not totally clear why this might be happening..

```
startxfce4

Cannot open /dev/tty0 (Permission denied)
```



## CI/CD SUPPORT

We have set up a new VM, *rg-ci-workflow1*, which can be used to host Github runners that launch Slurm jobs on the novel architecture within the Rogues Gallery.

---

**Note:** This resource is currently under development.

---

### 17.1 Resources

- [Setting up Github self-hosted runners](#)



## ROGUES GALLERY WORKFLOWS

*This page is currently a work in progress!*

Novel and post-Moore architectures typically have a specific workflow that incorporate some aspect of simulation or “emulation” before running on actual hardware. In a sense, this is because novel architectures are typically challenging to program or have some beta features that may crash when running a full program.

This page provides a guide for the general “suggested” workflows to be productive on the Rogues Gallery. Most of these workflows incorporate the usage of our Slurm job scheduler to request and use physical resources like servers and FPGAs.

### 18.1 Arm HPC

### 18.2 Reconfigurable - Xilinx

### 18.3 Reconfigurable - Intel

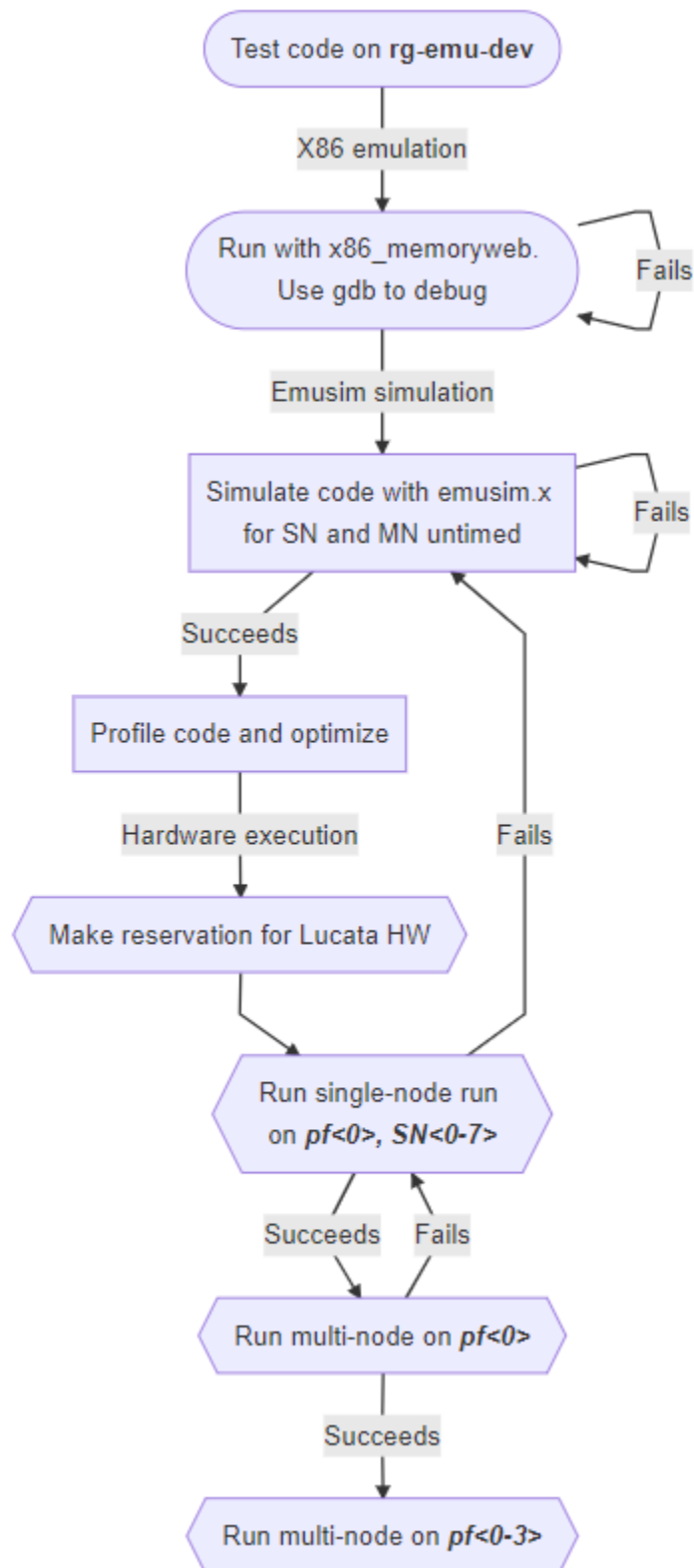
### 18.4 Bluefield Networking

### 18.5 Lucata Pathfinder

When getting started, we highly recommend checking out the [Lucata Pathfinder Programming Manual](#) (*requires GT Github login*) and read through Chapters 1,2,3, 5.1, 6, and 7. This will give you a basic understanding of the Cilk-based workflow and Lucata-specific APIs and tools.

As shown in the figure above, the suggested Lucata workflow combines 1) x86 functionality testing, 2) simulation of code on a VM, 3) execution on a single node of the Pathfinder system, and 4) execution on multiple nodes and chassis.

1. Compile your code on rg-emud-dev using <memoryweb.h> and emu-cc.sh to target x86 execution. This will run the Cilk code and emulate any data allocations specified by the Lucata APIs.
2. Simulate code on rg-emu-dev. Do debugging and initial verification here but note that simulation is slow! If you need to use a machine with more memory you can use [hawksbill.crnch@gatech.edu](mailto:hawksbill.crnch@gatech.edu)
3. Profile your code with the simulator for small input sets.
4. Make a reservation on the Google Calendar for the Pathfinder to run jobs. We also use our Slack channel to reserve time on the Pathfinder
5. Run your job on a single Pathfinder node (SN<0-7>. Verify its correctness.
6. Run your job on a single Pathfinder chassis (8 nodes PF<0-1>).





7. Run your job on multiple Pathfinder chassis (2 chassis).

## 18.6 Neuromorphic

## 18.7 Quantum



## PROFILERS

Profiling code is an important part of the optimization process for running new applications on novel architectures. While profilers typically target more mature architectures, there are a wide variety of Linux-based tools that can be used to evaluate the performance of your code, find bottlenecks, and optimize the execution of your code.

### 19.1 HPCToolkit

TBD

### 19.2 NVIDIA Nsight

TBD

### 19.3 Intel's VTune

Intel's VTune can be used for CPU, FPGA, and Intel GPU platforms along with MPI-based codes. See more information on using vTune [here](#).

### 19.4 Likwid

TBD

### 19.5 PAPI

TBD

## 19.6 Perf

TBD

## INSTINCT - AMD MI210

### 20.1 Acknowledgments

We greatly appreciate the donation of this test platform from AMD to [Dr. Spencer Bryngelson](#), a CRNCH faculty member working on computational fluid dynamics (CFD) codes including [MFC](#).

### 20.2 Current Status

#### 20.2.1 BUGS / Feature Requests

- N/A

### 20.3 System Specifications

Queue	CPU	Memory (GB)	Network	Cards	Notes
rg-gpu	2x <a href="#">AMD EPYC 7713 (Milan)</a>	512 GB DDR4, 3200 MHz, 32 GB DIMMs	Connect-X (MT28908), 10 GE	6 2x <a href="#">MI210</a>	

Instinct is a single node server with two AMD 210 GPUs, which are very similar to the MI250x GPUs deployed in [ORNL's Frontier](#). This server has two Milan CPUs, 512 GB of DDR4 memory, and a Connect-X 6 networking card.

### 20.4 Software and Tools

Distro	Kernel	Standard Compilers	Other Compilers	MPI	Miscellaneous
RHEL8	4.18.0	GCC 8.5	ROCm 5.6.0, <a href="#">AOCC 4.0.0</a>		AOCL 3.2.0, <a href="#">ROCm libraries 5.6.0</a> , <a href="#">uProf 4.0</a>

## 20.5 How do I get to Instinct?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

To request an allocation on Instinct using slurm:

```
//Request an allocation of 1 hr, partition rg-gpu, and specify the node name for the
↪server with -w
salloc -t 1:00:00 -p rg-gpu -w instinct
```

To request an entire node (and all the memory) you can run:

```
//Request an allocation of with 2 sockets, all cores in each socket, and 2 threads per
↪core, partition rg-gpu,
//specify the node name for the server with -w, and request unlimited memory (otherwise
↪cgroups limits to 1 GB per core)
salloc --sockets-per-node=2 --cores-per-socket=64 --threads-per-core=2 -p rg-gpu -w
↪instinct --mem=0
```

### 20.5.1 Compiling for the MI210 GPUs

Currently we suggest using the ROCm compiler stack to compile or GCC 12+.

Modules should be available but you may need to update your module path:

```
[$ module avail
----- /projects/tools/x86_64/rhel-8/modulefiles -
↪-----
aocc/3.2.0    aocl/aocc-3.2.0    aocl/gcc-3.2.0 (D)    rocm/5.6.0

[$ module load rocm/5.6.0
[$ which hipcc
/opt/rocm-5.6.0/bin/hipcc
```

Useful ROCm tools include hipcc, amdclang, aompcc, amdflang, rocdbg, and rocprof.

### Running OpenACC with GCC13

Alex Woods has compiled GCC13 for testing OpenACC on the Instinct GPU, and we have put his script in the internal docs [here](#) When compiling with the new gcc install, add the flags `-fopenacc -foffload-options=-march=gfx90a` for the Instinct MI2XX GPUs. Note that performance will likely be slow.

### 20.5.2 Running jobs

Once you have compiled your code, you can request a longer job to do testing.

### 20.5.3 Useful training material

- [OmniTrace](#) - AMD's newest tracing and profiling tool.
- [Introduction to Profiling Tools for AMD Hardware](#) - this article has a great breakdown of when to use specific profilers. OmniTrace and uProf are the newest profiling tools.

### 20.5.4 Vendor-provided Documents and Resources

- [AMD Training Center](#) (Getting Started, pointers to docs)





## QUORRA - NVIDIA AMPERE GPUS

### 21.1 Acknowledgments

Quorra1 is also funded via GT's Techfee program, so usage for this server is reserved for coursework (when requested by instructors or students). Please read more about Techfee hardware [on this page](#)

### 21.2 Current Status

#### 21.2.1 BUGS / Feature Requests

- N/A

### 21.3 System Specifications

Queue	Node Name	CPU	Memory (GB)	Network	Cards	Notes
rg-gpu	quorra1	2x AMD EPYC 7502 (Rome)	256 GB DDR4, 3200 MHz, 16 GB DIMMs	Connect-X 5 (MT27800), 100 GE; Bluefield-2 DPU (MT42822)	4x A30; 1x H100	
rg-gpu	quorra2	2x AMD EPYC 7502 (Rome)	256 GB DDR4, 3200 MHz, 16 GB DIMMs	Connect-X 5 (MT27800), 100 GE; Bluefield-2 DPU (MT42822)	4x A30; 1x A100	

### 21.4 Software and Tools

Distro	Kernel	Standard ers	Compil- ers	Other Compilers	MPI	Miscella- neous
Ubuntu 20.04	5.4.0	GCC 7.5		NVIDIA HPC SDK 23.3, CUDA 12.0		DOCA 1.5.1

## 21.5 How do I get to Quorra?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

To request an allocation on Quorra using slurm:

```
//Request an allocation of 1 hr, partition rg-gpu, and specify the node name for the
server with -w
salloc -t 1:00:00 -p rg-gpu -w quorra2 -G 1
//SSH to the resource
ssh quorra2
```

To request just one GPU - specify the type like A30, A100, or H100. .. code:

```
salloc -GH100 -prg-gpu
```

The local modules for NVIDIA HPC SDK should load automatically. However if they do not you can run “. /etc/profile.d/y02\_rg\_local\_modules.sh” to pull in all the local modulepaths.

## 21.6 Using NVIDIA SDK

```
module load nvhpc/23.3
```

### 21.6.1 MIG

See our internal docs for more instructions on using MIG with these GPUs [here](#).

### 21.6.2 Useful training material

TBD

### 21.6.3 Vendor-provided Documents and Resources

## VIOLET - SAPPHIRE RAPIDS

### 22.1 What's Interesting About This Hardware?

The Violet servers use Intel Sapphire Rapids along with PCI Express 5.0 to support high-bandwidth accelerators as well as the new Compute Xpress Link (CXL) technology. Violet1 also hosts CRNCH's H100 PCIe GPU in partnership with Dr. Bryngelson's lab.

### 22.2 Current Status

#### 22.2.1 BUGS / Feature Requests

- N/A

### 22.3 System Specifications

Queue	CPU	Memory (GB)	Network	Cards	Notes
rg-hpc	2x Intel 6454S (Sapphire Rapids)	512 GB DDR5, 4800 MHz, 32 GB DIMMs	Connect-X (MT28908), GE	6 1x NVIDIA H100 GPU 10	

Instinct is a single node server with two AMD 210 GPUs, which are very similar to the MI250x GPUs deployed in ORNL's Frontier. This server has two Milan CPUs, 512 GB of DDR4 memory, and a Connect-X 6 networking card.

### 22.4 Software and Tools

Dis-tro	Ker-nel	Standard Com-pilers	Other Compilers	MPI	Miscella-neous
RHEL8	4.18.0	GCC 8.5	Intel OneAPI 2023.0; NVIDIA HPC SDK 23.5; CUDA 12.0/11.0		

## 22.5 How do I get to Violet?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

To request an allocation on Violet using slurm:

```
//Request an allocation of 1 hr, partition rg-hpc, and specify the node name for the
↪server with -w
salloc -t 1:00:00 -p rg-hpc -w violet1
```

To request an entire node (and all the memory) you can run:

```
//Request an allocation of with 2 sockets, all cores in each socket, and 2 threads per
↪core, partition rg-hpc,
//specify the node name for the server with -w, and request unlimited memory (otherwise
↪cgroups limits to 1 GB per core)
salloc --sockets-per-node=2 --cores-per-socket=64 --threads-per-core=2 -p rg-hpc -w
↪violet1 --mem=0
```

### 22.5.1 Compiling for the Violet SPR CPUs

TBD

### 22.5.2 Vendor-provided Documents and Resources

- [Intel Xeon CPU Max Series Configuration and Tuning Guide](#) - Intel's tuning guide for this CPU architecture

## OCTAVIUS - A64FX TESTBED

*Last Updated: 10/14/22*

### 23.1 Current Status

- Octavius has been updated with the latest Cray Programming Environment and Lmod support.
- Note that we no longer have a “login” node, octavius-login. Please just submit jobs from rg-login!

#### 23.1.1 BUGS / Feature Requests

- No topology file for Slurm

### 23.2 System Specifications and Tools

Queues	CPU	Memory (GB)	Networking	Storage (GB)
rg-arm-debug, rg-arm-short, rg-arm-long	A64FX	32 HBM2e	EDR IB	330

Octavius is a 16 node cluster, where each node contains one CPU with integrated HBM memory, a small amount of SSD storage (340 GB per node), and a PCIe3 attached EDR InfiniBand card. The specific chassis for each node is the HPE Apollo 80, previously referred to as the Cray NSP-1 or “Baymax” platform. While this system is similar to standard Fujitsu servers, the main difference between these nodes and the Fugaku supercomputer is that Fugaku also includes a 6D Torus network that is tied into each CPU core.

#### 23.2.1 Current Tools

Note that the Cray compilers are likely to have the best support for SVE, although GCC 11 and 12 add more low-level support geared towards A64FX.

Distro	Ker- nel	Standard Compilers	Other Compilers	MPI	Miscella- neous
Cen- tOS 8	4.18.0	CCE 12.0.1, GCC 8.3, 9, 11.2.0	Armclang 22.0.2, Forge 22.0, Clang 10	MVAPICH2, Open- MPI 4.0	

**Debuggers:** gdb4hpc 4.7.2, 4.8.1; valgrind4hpc 2.7.1, 2.8.1

**Profilers:** PAPI 6.0, perftools 20.10.0

To use the latest Arm 22.0.2 compilers, you may need to update your modulepath (this will be fixed in future upgrades):

```
$ module use /net/projects/tools/aarch64/rhel-8/arm-allinea/arm-compiler/
↪22.0.2/modulefiles
Alternatively: $ export MODULEPATH=$MODULEPATH:/net/projects/tools/aarch64/rhel-8/arm-
↪allinea/arm-compiler/22.0.2/modulefiles
```

### 23.2.2 Tools - Arm Instruction Emulator

ArmIE is a DynamoRio front-end that allows for testing and evaluating SVE instructions on hardware that may not support the latest Arm ISA. As an example, you can use ArmIE to instrument your code with “regions of interest” and then print out instruction counts or the numbers and types of different operations (e.g., gather/scatter or load/stores).

For more information on using ArmIE, see this [Getting Started page](#) and this [tutorial page](#).

## 23.3 How do I get to Octavius?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

The figure below shows the basic outline of all the resources that you might interact with on the Arm cluster. Your network shared home is available on all nodes listed in this figure, although long compilations and tar/untar operations are suggested to use local scratch space on VMs or nodes.

The Mellanox SB7890 provides an InfiniBand connection to all the nodes within this cluster as well as to flubber3.

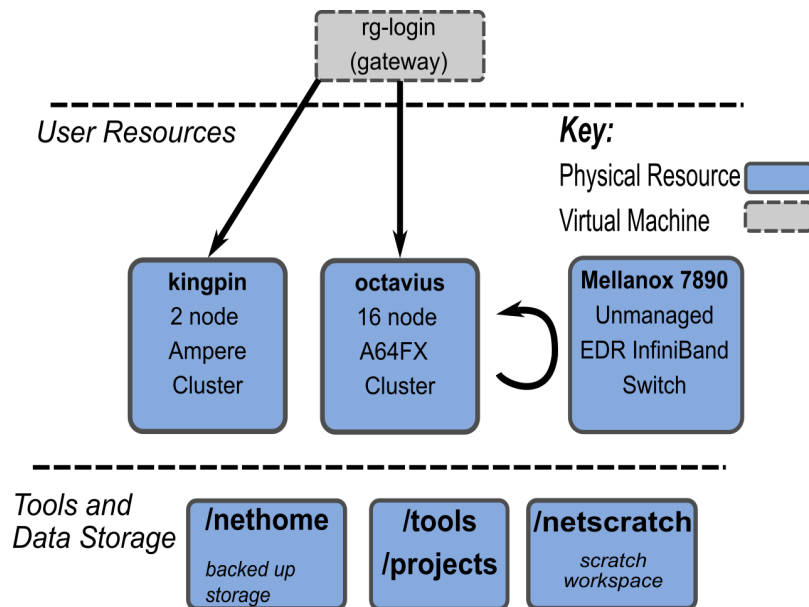


Fig. 1: RG Arm Server Overview

## 23.4 Compiling for the A64FX

Our installation includes a license for the Cray PE toolset, so you are encouraged to use this framework to compile your code. Clang and GCC are also currently available for compilation on the compute nodes.

To compile, you must request a short job (arm-debug queue) since we can't currently cross-compile on the x86 login node. By default, these jobs last an hour and are meant to be used to do compilation and basic testing.

For example:

```
[rg-login]$ sinfo
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
rg-arm-debug* up 2-12:00:00    4   idle octavius[1-4]
rg-arm-short  up 5-00:00:00   16   idle octavius[1-16]
rg-arm-long   up 15-00:00:0    16   idle octavius[1-16]

#Request an allocation for one hour
[rg-login]$ salloc -q rg-arm-debug -t 01:00:00 -N 1 -n 48 --exclusive
salloc: Granted job allocation 6

Check that your job has launched.
[rg-login]$ squeue
JOBID PARTITION    NAME      USER  ST        TIME  NODES NODELIST(REASON)
6 rg-arm-de      bash  gburdell  R         0:21      1 octavius1

#Launch an interactive job, run your compilation and testing, and then exit the node
#Note that the -l flag indicates that the interactive job should load local environment
↳scripts and is needed!
[rg-login]$ srun --pty bash -l
gburdell@octavius1:~$ <run_compilation>
gburdell@octavius1:~$ exit

#Cancel your job if needed (ie, if you have just used a few minutes)
[rg-login]$ scancel 6
[rg-login]:~$$ salloc: Job allocation 6 has been revoked.
```

To request a specific node you can use the `-w <nodename>` flag

```
[rg-login]$ salloc -q rg-arm-debug -t 01:00:00 -N 1 -n 48 --exclusive -w octavius2
[rg-login]$ squeue
          JOBID PARTITION    NAME      USER  ST        TIME  NODES NODELIST(REASON)
          203 rg-arm-de      bash  gburdell  R         0:04      1 octavius2
//srun and execute job
```

## 23.5 Using modules

By default octavius uses LMOD. Use the standard module avail and module load commands. If you don't see all the modules you can try to source our script which will update your MODULEPATH environment variable.

```
//Shows how to source this script to add all available module paths
$ . /tools/armhpc/init_modules_slurm_octavius_manual.sh
$ module avail
```

This would produce the following output (elided for clarity)::

```
----- /opt/cray/pe/perftools/default/
↪modulefiles -----
perftools-lite-events    perftools-lite-gpu    perftools-lite-hbm    perftools-lite-
↪loops    perftools-lite    perftools-preload    perftools
----- Cray Modules -----
↪-----
PrgEnv-cray/8.1.0        (L,D)    cray-hdf5/1.12.0.2        (D)    ↪
↪cray-parallel-netcdf/1.12.1.0 (D)

----- /opt/cray/pe/craype-targets/
↪default/modulefiles -----
craype-arm-nsp1    craype-arm-thunderx2    craype-network-infiniband

----- Octavius Modules -----
↪-----
cmake/3.21.3    gnu9/9.4.0    hwloc/2.5.0    libfabric/1.13.0    openmpi4-gnu9-backup/4.
↪0.4    prun/2.2    ucx/1.11.2

----- Arm Compilers -----
↪-----
acfl/22.0.2    armie22/22.0    binutils/11.2.0    gnu/11.2.0    gnurt11/11.2.0

----- /net/projects/tools/aarch64/rhel-
↪8/modulefiles -----
arm-forge/22.0.2    likwid/5.1.1
```

## 23.6 Running jobs

Once you have compiled your code, you can request a longer job to do testing.

```
# Clone the Slurm examples from our internal wiki
$ git clone https://github.gatech.edu/crnch-rg/rogues-docs.git
$ cd slurm_examples
$ sbatch octavius_cpe_mvapich.sbatch
Submitted batch job 539
```



## 23.7 Requesting new packages or assistance

Please just [submit a ticket](#) or ask on our Teams group in the “help-request” channel. We also have a general arm-hpc discussion channel on our Teams group.

## 23.8 Useful training material

The recent [SVE tutorial](#) by Arm is probably the best source for learning how to use SVE with A64FX. You can also ask questions in the hackathon channel on the [Arm HPC User Group Slack](#).

[2021 SVE Hackathon for Ookami](#)

[SVE Hackathon repo](#) -also available under `/tools/training/arm-hpc` on RG nodes

## 23.9 Vendor-provided Documents and Resources

- [Cray PE Programming Guide \(10/20\)](#) - a copy of this is also included in the rogues-docs repo.
- [Arm A64FX Architecture Manual](#)
- [Arm wiki for optimizing HPL](#)
- [Cray Programming Environment Workshop Slides from NERSC](#)



## KINGPIN - NVIDIA DEVKIT SYSTEMS

### 24.1 Acknowledgments

The Kingpin nodes are hoteled hardware sponsored by the [Habanero research group](#) led by Dr. Sarkar. This means this group has priority on this hardware via a separate queue but it can be used when not in use by this group.

### 24.2 Current Status

#### 24.2.1 BUGS / Feature Requests

- N/A

### 24.3 System Specifications

Queues	Node Name	CPU	Memory (GB)	Network	Cards	Notes
	kingpin1					

### 24.4 Software and Tools

Distro	Kernel	Standard Compilers	Other Compilers	MPI	Miscellaneous
RHEL 8	5.4.0	GCC 7.5	NVIDIA HPC SDK 23.3, CUDA 12.0		DOCA 1.5.1

## 24.5 How do I get to the Kingpin nodes?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

To request an allocation on Kingpin using slurm:

```
//Request an allocation of 1 hr, partition rg-hpc, and specify the node name for the
↪server with -w
salloc -t 1:00:00 -p rg-hpc -w kingpin1
```

## 24.6 Using NVIDIA SDK

```
module load nvhpc/23.5
```

## LUCATA PATHFINDER GETTING STARTED



The Rogues Gallery hosts two distinct systems from Lucata (formerly known as Emu Technology): The Gen1 Emu Chick, an 8-node desktop-style system, and the Lucata Pathfinder, a four-chassis system with 16 nodes and 24 cores in each node for a total of 768 cores across the Pathfinder cluster.

## 25.1 Latest Updates

- The 23.07 Pathfinder tools have been released and made the default on the Rogues Gallery testbed. See [the 3.3 Programming Guide here](#) (RG user account login to GT Github required) for the latest information on this toolset release.

## 25.2 Suggested Background and First Steps

We suggest that you familiarize yourself with the Cilk programming language and Cilk syntax using the [OpenCilk tutorials](#).

## 25.3 Using the EMU simulation and compiler tools

The current toolset, documentation, and examples are available on the rg-emu-dev VM and other nodes as a module. Note that it is recommended to use the latest version of the Pathfinder toolset.

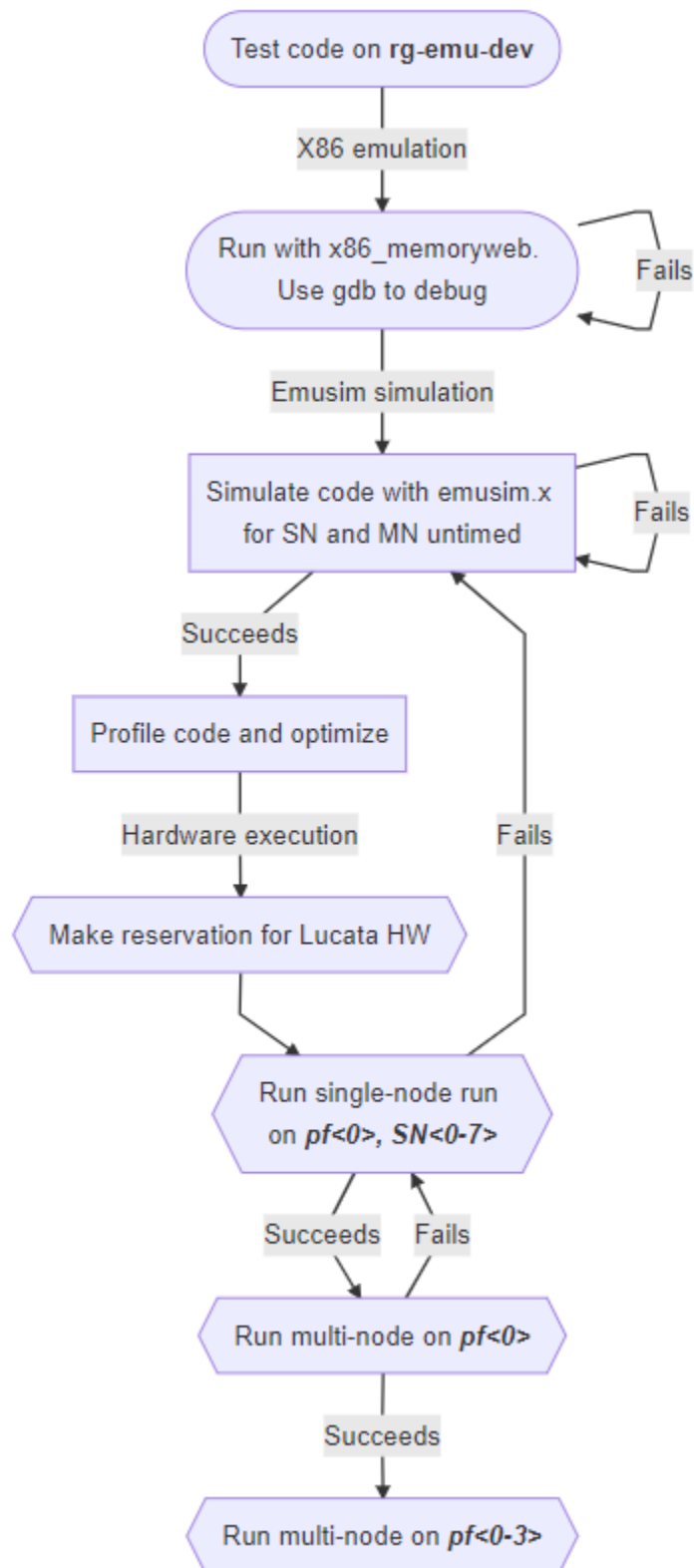
- **rg-login.crnch.gatech.edu**: primary login VM for Rogue's Gallery. Use this VM to log in to another node for testing and simulation from off campus.
- **rg-emu-dev.crnch.gatech.edu**: VM for **Lucata compilation and simulation**
- **hawksbill.crnch.gatech.edu**: Large-memory, general-purpose node for larger Lucata-based simulations
- **pf<0-3>.crnch.gatech.edu**: Lucata Pathfinder chassis for HW execution

## 25.4 Lucata Workflow

When getting started, we highly recommend checking out the [Lucata Pathfinder Programming Manual](#) (*requires GT Github login*) and read through Chapters 1,2,3, 5.1, 6, and 7. This will give you a basic understanding of the Cilk-based workflow and Lucata-specific APIs and tools.

As shown in the figure above, the suggested Lucata workflow combines 1) x86 functionality testing, 2) simulation of code on a VM, 3) execution on a single node of the Pathfinder system, and 4) execution on multiple nodes and chassis.

1. Compile your code on rg-emu-dev using <memoryweb.h> and emu-cc.sh to target x86 execution. This will run the Cilk code and emulate any data allocations specified by the Lucata APIs.
2. Simulate code on rg-emu-dev. Do debugging and initial verification here but note that simulation is slow! If you need to use a machine with more memory you can use hawksbill.crnch.gatech.edu
3. Profile your code with the simulator for small input sets.
4. Make a reservation on the Google Calendar for the Pathfinder to run jobs. We also use our Slack channel to reserve time on the Pathfinder
5. Run your job on a single Pathfinder node (**SN<0-7>**). Verify its correctness.
6. Run your job on a single Pathfinder chassis (8 nodes PF<0-1>).
7. Run your job on multiple Pathfinder chassis (2 chassis).



## 25.5 Tutorials and Training

Please check out the recent [Pathfinder tutorial](#) for official training material for the Pathfinder systems. There are also some examples and related tools shared in a Github repo at <https://github.gatech.edu/crnch-rg/emu-common> (requires login). Please feel free to branch and fork as makes sense for your research.

Eric Hein has also contributed a nice micro-benchmark that uses serial and recursive spawn. [Micro benchmark](#)

## 25.6 Other resources:

- The [GraphBLAS branch](#) can be found [here](#)
- CilkPlus can also be run on CPU-based clusters. For more information on general CilkPlus check out the official [website](#) and other [Cilk tutorials](#).
- See our Kokkos [branch](#) focused on CilkPlus and eventually on an Emu backend. For more information on Kokkos, check out their [website](#), [tutorials](#), and other documentation.



## LUCATA PATHFINDER FAQs

This page details answers to some commonly asked questions about the Lucata Pathfinder system.

### 26.1 What are the detailed specs for GT's Pathfinder systems?

GT currently hosts a four-chassis Pathfinder system which has 32 nodes in total with 8 in each chassis. The nodes are connected directly via RapidIO links.

- Stationary cores per node/chassis: 1/8
- Memory-side processors per node/chassis: 8/64
- Memory channel width: 16 bit
- Memory per node/chassis: 64 GB/512 GB
- Gossamer Cores or Lucata Compute Elements (LCE) per node/chassis: 24/192
- LCE clock speed: ~220 MHz (may be slightly higher/lower)
- Maximum number of concurrent threads per node/chassis: 1536/12,288 (additional spawned threads sit in the run queue)
- Power usage per chassis: ~2KW max although likely lower in practice
- System node CPUs are PowerPC e6500 T2080 CPUs with 8 cores, running at 1.8 GHz
- **Nodes inside a chassis are connected to 6 internal switches that each have 12 ports**
  - 4 ports per switch can be used to connect to other chassis while other ports connect to all 8 nodes in an endpoint star formation
  - Within a chassis each node should be accessible within 1 switch hop; 2 chassis requires 2 switch hops
  - Note that GT does not have a rack-mounted switch that would further scale the network.



## RUDI2 - JETSON AGX ORIN

### 27.1 What's Interesting About This Hardware?

### 27.2 Current Status

#### 27.2.1 BUGS / Feature Requests

- N/A

### 27.3 System Specifications

Queues	CPU	Memory (GB)	Network	Cards	Notes
rg-neuro					

### 27.4 Software and Tools

Distro	Kernel	Standard Compilers	Other Compilers	MPI	Miscellaneous
Ubuntu 20.04	4.18.0	GCC 8.5	Intel OneAPI 2023.0; NVIDIA HPC SDK 23.5; CUDA 12.0/11.0		

### 27.5 How do I get to rudi2?

As with most CRNCH resources, you need to either log in via the gateway node, rg-login, or access the system from the campus network via VPN or an on-campus connection.

To request an allocation on rudi2 using slurm:

```
//Request an allocation of 1 hr, partition rg-neuro, and specify the node name for the server with -w
salloc -t 1:00:00 -p rg-neuro -w rudi2
```

Using NVIDIA Jetson Developer Kit Modules ——— - `NVIDIA Jetson Developer Kit User Guide

## 27.5.1 Vendor-provided Documents and Resources

## SMART NETWORKING - GETTING STARTED

The Rogues Gallery supports three types of novel networking:

- FPGA-based network devices
- CPU-based smart network devices
- 5G networking testbed equipment

### 28.1 Acknowledgments

We appreciate the donation of several Bluefield-2 boards from NVIDIA.

### 28.2 Smart Networking Hardware

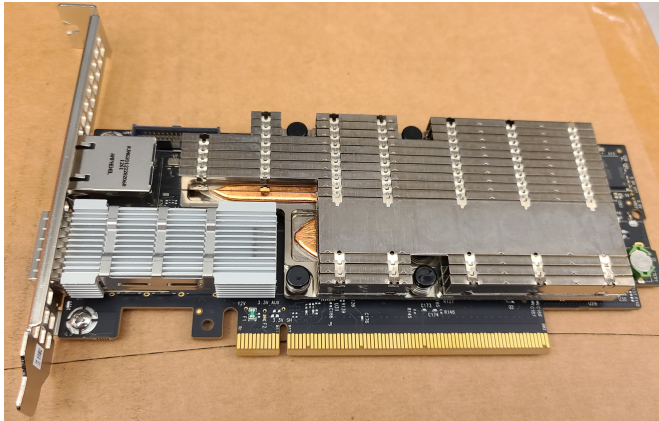
Currently, the Rogues Gallery hosts the following smart networking hardware.

Table 1: **Smart Networking Hardware**

Networking Accelerator	Processor Type	Memory	Hosting Machine	Notes
Bluefield-1 DPU			flubber6/7	
Bluefield-2 DPU	Arm CPU	A72 16 GB DRAM	quorra1/2, flubber6/7, king-pin1/2, flubber10	1 board in most nodes, 4 in flubber10
InnovaFlex-2			flubber6/7	
Intel N6000			flubber6/7	
Xilinx Alveo U50			flubber1	
Xilinx Alveo U280			flubber5	
5G equipment (TBD)				



## BLUEFIELD-2 DPU



NVIDIA's Bluefield-2 is the second generation of their Data Processing Unit (DPU) card which has both a high-speed InfiniBand/Ethernet interconnect and on-board DRAM and Arm CPU cores. Notably, this is one of the first SmartNIC devices that can run an on-board operating system in “host” mode.

---

**Note:** These resources are currently being deployed so please check with the testbed maintainers on Slack and Teams to get started!

---

### 29.1 Bluefield-2 Cards

Resource	Cards	Notes
quorra1-bf2-mgt	Bluefield-2 NIC	
quorra2-bf2-mgt	Bluefield-2 NIC	





## CUQUANTUM

This page details how to use [cuQuantum](#) on NVIDIA GPUs within the CRNCH Rogues Gallery. cuQuantum specifically is focused on accelerating state vector and tensor network simulations while QODA is a higher-level framework and API that is targeted toward hybrid quantum-classical computing (think like a CPU+QPU rather than a CPU+GPU).

- [NERSC 2022 cuQuantum and QODA Presentation](#)



## XILINX FPGAS - GETTING STARTED

### 31.1 Acknowledgments

We appreciate the donation of 2x Alveo U280 boards and licenses from Xilinx's University Program as well as the donation of an AC-510 and AC-511 HMC chip from Micron.

### 31.2 Xilinx FPGA Hardware

Currently, the Rogues Gallery hosts the following Xilinx FPGA hardware.

Table 1: **Server-based Hardware**

FPGA Board	FPGA Chip	Memory	Hosting Machine	Notes
Xilinx Alveo U50		8 GB HBM	flubber1	2 boards
Xilinx Alveo U250			flubber9	
Xilinx Alveo U280			flubber<4-5>	1 board; 2 boards
Xilinx SmartSSD			flubber9	
AC-510	XCVU060 (Pkg FFVA1156)	4 GB HMC 1.0	rg-fpga-cubed	

Table 2: **Development Board Hardware**

FPGA Board	FPGA Chip	Memory	Hosting Machine	Notes
ZCU102				
Versal VCK190		TBD		
Pynq Z2			synestia2	50+ boards available for classes

### 31.3 Documentation and Related Repositories

- All of our manuals for Alveo boards can be found in the [reconfig-docs repo](#) (requires RG login).

## 31.4 Accessing the Rogues Gallery Xilinx FPGAs

See the [Xilinx FPGA Workflow](#) page here for more details on the process to use this hardware.

The short version is:

- For emulation and development, please use the FPGA development VMs, *rg-fpga-dev-1-4*.
- For GUI-based development, use VNC, ideally with either Slurm or our [OOD instance](#).
- For final bitstream compilation, request a hardware node using Slurm and run the tools on that node.

## 31.5 Xilinx Software

Please see the specific pages on using [Vitis \(standard flow\)](#), [Vivado flow \(advanced\)](#), or frameworks for SmartNics like OpenNIC.

### 31.5.1 Vitis vs. Vivado vs..

Xilinx released the Vitis SW and HW focused framework officially in 2020 to complement existing tools like Vivado (for RTL-based designs), and replace existing tools like SDx (for OpenCL-based designs). However, it is still somewhat confusing as to which tool you might want to use for which situation. This [issue on the Vitis Tutorials repo](#) includes an insightful table from Víctor Vilches which we have included and modified slightly.

Table 3: Xilinx Tool Variations (from Issue 73 and Víctor Vilches)

Tool	Stakeholder	Aim of Tool
Vivado	Hardware engineer	Develops RTL (ie, Verilog) kernels
Vitis	Embedded engineer	Creates low level firmware, BSPs, boot sequences and integrates software and hardware efforts
Vitis HLS	Software engineer	Develops C, C++ or OpenCL kernels
Vitis AI	Data scientist	Uses HLS and frameworks to develop AI constructs/kernels

### 31.5.2 Supported Versions of Xilinx Tools

With that in mind, we support the following versions of Xilinx software on our testbed:

Table 4: **Supported Software**

Software	Versions	Notes
Vitis	2020.2, 2021.1, 2021.2, 2022.1	
Vivado	2020.2, 2021.1, 2021.2, 2022.1	
Vitis HLS	2020.2, 2021.1, 2021.2, 2022.1	
Vitis-AI		
SDAccel	2019.2	Deprecated

- How to quickly start Vitis/Vivado 20.2:

```
export XILINXTOOLSHARE=/tools/reconfig/xilinx
#Source the different Vivado/Vitis settings files
. $XILINXTOOLSHARE/Vivado/2020.2/settings64.sh
. $XILINXTOOLSHARE/Vitis/2020.2/settings64.sh
. $XILINXTOOLSHARE/Vitis_HLS/2020.2/settings64.sh
```

### 31.5.3 How do I check the licenses that are available?

You can either use the licensing center from the GUI version of a tool like Vivado or Vitis, or you can run the following:

```
lmutil lmstat -a -c 2100@toolbox.crnch.gatech.edu

lmutil - Copyright (c) 1989-2016 Flexera Software LLC. All Rights Reserved.
Flexible License Manager status on Mon 4/5/2021 21:37

License server status: 2100@toolbox
  License file(s) on toolbox: /tools/reconfig/licenses/xilinx_vivado_july2020.lic:

toolbox: license server UP (MASTER) v11.14.1

Vendor daemon status (on toolbox):

  xilinx: UP v11.14.1

Feature usage info:

Users of Vivado_System_Edition: (Total of 15 licenses issued; Total of 0 licenses in use)
...
Users of SDK: (Total of 15 licenses issued; Total of 0 licenses in use)
...
Users of HLS: (Total of 15 licenses issued; Total of 0 licenses in use)
...
```

### 31.5.4 Getting started with AWS for development

Amazon supports F1 instances that have between 1 and 8 Xilinx FPGAs. Currently they support the VCU1525 with an Ultrascale+ part. Xilinx and Amazon both have good references on getting started with these instances.

- [Xilinx Getting Started with AWS](#)
- [SDAccel AWS examples from Accelerator Program](#)
- [SDAccel AWS Labs](#)

### 31.5.5 Xilinx Accelerator Program

Xilinx has a program for faculty and staff that seems to provide discounts on Alveo board (~\$1500 discount for up to two board) and links to existing resources for SDAccel, AWS, and Alveo products. These are normally behind a login wall, but please see the links below:

- [Alveo Getting Started](#)
- [Alveo FPGA Tutorial](#) geared towards Nimbix (another cloud provider)
- [General Xilinx Forums](#)

### 31.5.6 Xilinx Machine Learning Options

- [Xilinx ML page](#) (TBD)

## XILINX VIVADO FLOW

This page describes using the “Vivado Flow” with Alveo boards as well as other platforms in the Rogues Gallery. Note that this is considered advanced because it can cause the boards to lock up.

Most of the relevant documentation for “custom” flows can be found at this [Xilinx support link](#).

### VERY IMPORTANT NOTE

The Alveo boards are mostly restricted to Vitis programming flow in other centers. It is IMPERATIVE that you do not program HBM-enabled Alveo boards incorrectly as they will need to be RMAed to be fixed. This is ultimately a Xilinx “documentation bug”, but we have had this happen to 2-3 of our cards which causes long downtimes

This bug is described in [Support Article 72926](#). The important note is that you need to follow the 2 steps listed:

1. In the Vivado design, connect pin D32 to the CATTRIP output from the HBM IP or connect to GND
2. Add the following constraints (do not include the pulldown constraint if connecting to the CATTRIP output from the HBM IP):

```
set_property PACKAGE_PIN D32 [get_ports "HBM_CATTRIP"];# Bank 75 VCC0 -u
↪VCC1V8 - IO_L17P_T2U_N8_AD10P_75
set_property IOSTANDARD LVCMOS18 [get_ports "HBM_CATTRIP"];# Bank 75 VCC0 -u
↪VCC1V8 - IO_L17P_T2U_N8_AD10P_75
set_property PULLDOWN TRUE [get_ports "HBM_CATTRIP"];
```

If you do happen to fail to follow these instructions, we can try the steps on the [Alveo Debugging Page](#). However, we have had to RMA several of our U280s in the past 1-2 years due to this bug and user error.

## 32.1 Alveo Documentation

We have moved the Alveo documentation, XDC (constraint files), and related tools into a separate repo on our GT repo called [reconfig-docs](#). Please log in using your Rogues Gallery username and password.

### 32.1.1 General Locations of Board Files

Xilinx now posts their board files in a Github repos for the [Board Store](#) and [Vivado example designs](#).

#### U50 Boards

Vivado Flow requires a special Alveo programming cable that is attached via USB to the host server. This is detailed in the [Alveo Programming Cable Guide](#).

TBD - need links to the user guide and programming guide

TBD - need links to the HBM

U50 has a reference design but it's pretty large. If it's useful we should link it here.

#### U280 Boards

Vivado flow requires a USB programming cable to be connected into the card.

TBD - need links to the user guide and programming guide

TBD - need links to any HBM Guidance

TBD - how to check that the board shows up in Vivado.



## PYNQ CLUSTER



Xilinx's Python or PYNQ framework has recently been extended from supporting just PYNQ-branded boards to full devices like the Zynq and Alveo platforms. This means that PYNQ can be used on the Rogues Gallery either with PYNQ branded boards or with specific Zynq or Alveo boards.

## 33.1 PYNQ Cluster

The PYNQ cluster is a fully scheduled and remotely accessible PYNQ-Z2 cluster, which is part of the Rogues Gallery's educational outreach mission. As such they are on the *cc.gatech.edu* subnet, and they need to be accessed via the *synestia2* VM with Slurm.

Resource	CPU	Memory (GB)	FPGA Fabric	Software and Features
pynq-z2-<1-40>	2 core, Arm A9	0.5	85K cells	logic ZYNQ XC7Z020-1CLG400C (7000 series)
synestia<1-18>	4 core, E312xx	12	NA	Ubuntu 20.04 VM for FPGA tools

## 33.2 Accessing the PYNQ cluster

The most up-to-date documentation is currently on the CS3220 website. Eventually it will be updated and migrated to this page.

- To access the Synestia VMs see this link [https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/access\\_vm\\_steps/access\\_vm\\_doc.md](https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/access_vm_steps/access_vm_doc.md).
- To access an FPGA from Synestia, please see this link [https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/access\\_pynq\\_boards/updated\\_instructions.md](https://github.com/gt-cs3220/gt-cs3220.github.io/blob/master/access_pynq_boards/updated_instructions.md)

## 33.3 PYNQ Resources

- Alveo PYNQ Examples
- PYNQ Tutorial - FCCM 2020 Workshop
- Pypi PYNQ tutorial

## XILINX SMART SSD

The [Xilinx SmartSSD](#) provides an FPGA programmable interface to an NVMe SSD drive that can be used to accelerate storage-oriented workloads.

[Xilinx SmartSSD User Guide](#)

### 34.1 Smart SSD Getting Started

Right now, our Smart SSD is located in an NVMe slot on flubber9. It needs to be installed in a server since it can get quite hot and it requires an NVMe interface.

### 34.2 Current Firmware

This section is just for reference, as you should not need to update the firmware on this device.

We compiled the latest XRT from the Github repo so are running XRT 2.14. One of the Samsung XRT files was missing `/opt/xilinx/xrt/share/fw/sched_v30.bin` when attempting to install `xilinx-u2-gen3x4-xdma-gc-2-202110-1-dev-1-3311104.noarch`. We extracted the files from Samsung XRT debfiles and copied the file to our local XRT installation location and the errors were resolved.

```
#Extract Samsung XRT debfile and copy the missing file
$ rpm2cpio 'openDownload?filename=xrt_202120.2.12.427_8.1.1911-x86_64-xrt.rpm'
↪| cpio -idmv
$ cp opt/xilinx/xrt/share/fw/sched_v30.bin /opt/xilinx/xrt/share/fw

#Install the firmware for the SmartSSD
$ dnf -y install xilinx-u2-gen3x4-xdma-gc-2-202110-1-dev-1-3311104.noarch
↪xilinx-u2-gen3x4-xdma-gc-base-2-3311104.noarch.rpm xilinx-u2-gen3x4-xdma-gc-
↪validate-2-3311104.noarch.rpm
$ ls /opt/xilinx/firmware/u2/gen3x4-xdma-gc/base/
firmware  license  partition_metadata.json  partition.xsabin  scripts  test

#Find the right BDF to program
$ xbmgmt examine

Devices present
BDF          : Shell      Platform UUID  Device ID      Device Ready*
[0000:24:00.0] : xilinx_u2  0x0           mgmt(inst=9216) No
```

(continues on next page)

(continued from previous page)

```
#Program the device with the correct firmware
$ /opt/xilinx/xrt/bin/xbmgmt program --base --device 0000:24:00.0 --image /opt/
↪xilinx/firmware/u2/gen3x4-xdma-gc/base/partition.xsabin
```

After the firmware is programmed, a cold boot is required to load the new firmware image on the FPGA. Cold boot refers to fully power OFF the machine and then power it ON again.

### 34.2.1 What is the “flat” firmware used for?

See Appendix H of the User Guide <[https://www.xilinx.com/content/dam/xilinx/support/documents/boards\\_and\\_kits/accelerator-cards/1\\_3/ug1382-smartssd-csd.pdf](https://www.xilinx.com/content/dam/xilinx/support/documents/boards_and_kits/accelerator-cards/1_3/ug1382-smartssd-csd.pdf)>`\_\_ which refers to the layout of the flat shell, which allows for mixing the static “shell” (PCIe, DRAM controller) and dynamic user logic.

## VORTEX RISC-V GPGPU

The Vortex RISC-V GPGPU is a research project that is focused on developing a RISC-V ISA compatible GPGPU that can execute real code. The [Vortex site](#) has more information on the project including the software stack and recent extensions for executing graphics code. This page focuses on how to get started with using and testing Vortex on the Rogues Gallery.

We suggest that you pair the instructions here with the [Vortex tutorial](#).

### 35.1 Using Vortex with Rogues Gallery

You can run the Vortex toolchain and simulation on multiple nodes within RG including the *rg-fgpa-dev* VMs, Intel, and Xilinx flubber nodes. Note that the Xilinx port of Vortex is currently in progress, so here we focus on using Vortex with the Intel FPGA nodes (flubber<1-3>).

#### 35.1.1 Vortex Installation Dependencies

Normally we suggest that you use the prebuilt toolchain which includes a RISC-V cross-compiler.

For Ubuntu:

For RHEL8:

### 35.2 Vortex Software Toolchain

TBD



## XILINX ML TOOLS

We have been notified by Xilinx that [Vitis-AI](#) is the preferred framework going forward.

### 36.1 Vitis-AI

We installed [Vitis AI](#) using the Docker instructions and installing Docker on Ubuntu using the recommended [Docker Engine installation path](#).

Note that you must be part of the Docker group and download the Vitis AI repo to your local home directory or ideally to your localscratch directory.

Check your docker group status and that you can run Docker images without sudo.

```
$ groups | grep docker
gtguest docker ... coc-rg-login ...

#Test a basic Docker image - if this fails your group
#permissions are not correct; please submit a ticket
$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.
...
```

Run the docker image - **note you should not need to pull a new Docker image as this is shared for all users**

```
#Pull the Vitis AI repo, which includes their docker run script. It is suggested to make
↳ a directory on localscratch
$git clone --recurse-submodules https://github.com/Xilinx/Vitis-AI

#List the available images - note the common Vitis AI image for all users
$docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
xilinx/vitis-ai      latest      a7eb601784e9     4 months ago    10.1GB
hello-world         latest      bf756fb1ae65     10 months ago   13.3kB

#Switch to your Vitis-AI repo directory and run the docker startup script
$cd Vitis-AI
$./docker_run.sh xilinx/vitis-ai:latest
#Accept the terms for the container
```

(continues on next page)

(continued from previous page)

```

=====

--      --      --      --      --

\ \ / / ( ) | ( )      ^      | _ _ |
\ \ / / | | _ _ | _ _ | / ^ \ | | |
\ \ / / | | _ _ | \ _ \ / ^ \ | | |
\ \ / / | | \ _ \ | _ _ / / ^ \ | | |
\ \ / / | | \ _ \ | _ _ / / ^ \ | | |

=====

`Docker Image Version: latest
Build Date: Mon Nov 16 10:46:01 MST 2020
VAI_ROOT=/opt/vitis_ai
For TensorFlow Workflows do:
  conda activate vitis-ai-tensorflow
For Caffe Workflows do:
  conda activate vitis-ai-caffe
For Neptune Workflows do:
  conda activate vitis-ai-neptune
More detail on conda packages included in container: /opt/vitis_ai/conda/conda_packages.
↪txt
More detail on other 3rd party package source included in container: https://www.xilinx.
↪com/products/design-tools/guest-resources.html
flubber3:/workspace$`

```

## 36.2 Xilinx Support for TVM

- <https://sampl.cs.washington.edu/tvmconf/slides/2019/Elliott-Delaye-Xilinx.pdf>
- <https://xterra2.avnet.com/xilinx/ml/tvm-on-xilinx-mpsoc>

## 36.3 Xilinx DPU (Deep Processing Unit)

- <https://github.com/Xilinx/DPU-PYNQ>

## 36.4 Experimental Projects

- [FINN](#) - a Python and PyTorch-based framework



## 36.5 Legacy tools - Xilinx's MLSuite

Xilinx's MLSuite is currently described by Xilinx as a "legacy product". It supports mapping TensorFlow and Caffe models to FPGA hardware, and it has evolved from basic GEMM support to the current MLSuite, supported on AWS boards (VCU1525) and newer Alveo boards.

- [Xilinx MLSuite main page](#)
- [MLsuite Github](#) - contains the most resources and info on installation and usage of the tools.



## INTEL ONEAPI FOR RECONFIGURABLE COMPUTING

Intel has recently released their oneAPI framework for programming CPUs, GPUs, and FPGAs. oneAPI or Data-parallel C++ (DPC++) is a variant of the SYCL language with added features to better support Intel platforms. Note that much of the DPC++ ecosystem is built on LLVM, so related tools like HipSYCL and Xilinx's SYCL support *may* be compatible with DPC++ code (see [Xilinx's triSYCL for an example](#)).

### 37.1 How can I test out oneAPI for FPGAs?

There are several resources for using oneAPI both online and at Georgia Tech mentioned below. In general we suggest the following steps:

- 1) Get an account for the Rogues Gallery testbed or Intel's DevCloud.
- 2) Learn the basics of SYCL/One API via a tutorial or video series (see resources below).
- 3) Work through Tier 1 and 2 of the [OneAPI FPGA code samples tutorial](#). Learning about the basics up to loop unrolling is highly recommended.
- 4) Test out your own code, using the [Intel FPGA optimization guide](#) as a reference!

See more details on a workflow for testing out emulation and building bitstreams with RG platforms at this page (TBD).

### 37.2 Acknowledgments

We appreciate the donation of software licenses from Intel's University Program.

### 37.3 Intel FPGA Hardware

Currently, the Rogues Gallery hosts the following Intel FPGA hardware.

Table 1: Server-based Hardware

FPGA Board	FPGA Chip	Memory	Hosting Machine	Ma-	Notes
Bittware IA-840F	AGF027	128 GB DDR4	flubber8		
Stratix 10 PAC	GX2800	32 GB DDR4	flubber3		
Bittware 520N	GX2800	16 GB DDR4	flubber2		
Bittware 520N-MX	MX2100	16 GB HBM	flubber3		
Intel Arria10 PAC	GX1150	8GB DDR4	flubber2		
Bittware 385-A	GX1150	8 GB DDR3	flubber2		Shared via the GTRI Cipher lab
Bittware 385-SoC	SX660 (Arm+GX1150)	6 GB DDR4	flubber3		Shared via the GTRI CIPHER lab

Table 2: Development Board Hardware

FPGA Board	FPGA Chip	Memory	Hosting Machine	Notes
Intel Arria10 DevKit	GX1150	4 GB DDR3	NA	Available upon request

### 37.3.1 What tools are available on RG?

OneAPI is available as a [base toolkit](#), which includes tools like [vTune](#) and the [Application Performance Snapshot Tool](#), [Intel Advisor](#), and a [DCPP Compatibility tool](#) for migrating CUDA codes to DPC++.

There are also toolkits for [HPC](#), [Deep Learning Framework](#), [FPGA](#), and [OpenVino](#) that build on the base oneAPI toolkit.

Intel's [Open Fabric Stack](#) is supported for specific Intel-branded boards like the Arria 10 and Stratix 10 PAC cards in the Flubber nodes.

### 37.3.2 Online Tutorials

- [Intel's oneAPI and SYCL tutorials](#) - see the “Essentials of SYCL” and “FPGA Development Flow Using Intel oneAPI Base Toolkit”.
- [A general SYCL tutorial](#) - oneAPI is a variant of SYCL with some Intel-specific implementation features added.
- [SYCL tutorial from SYCLcon 2022](#)

### 37.3.3 Useful Documents

- [oneAPI Programming Guide](#) - a general reference for OneAPI and Intel's implementation of SYCL.
- [Intel FPGA Optimization Guide](#) - the key reference for optimizing code for Intel FPGAs
- [DPC++ book](#) - An Open Source Data Parallel C++ book
- [Intel Github OneAPI code samples](#)
- [Tech.Decoded webinars](#)

### 37.3.4 OneAPI Related Videos and Webinars

- [OneAPI Basics Training Series](#) - a General OneAPI series
- [Introduction to Intel's Open FPGA Stack](#) - an introduction to the Intel OFS.
- [Using FPGAs with the Intel oneAPI Toolkits](#)
- [Introduction to Optimizing FPGAs with the Intel OneAPI Toolkit](#)

### 37.3.5 Other Online OneAPI Resources

- [Docker containers](#) with the “base” toolkit
- [Intel DevCloud](#) - you can easily get a 3 month pass that can be extended by registering a project on the “Intel DevMesh”. Once you sign up, you can [connect](#) either via an SSH terminal (using a provided SSH config script) or using a JupyterHub notebook interface. There are workshop files that can be copied to your home directory using the following command: `/data/oneapi_workshop/get_oneapi_workshop.sh`



## USING INTEL ONEAPI WITH CRNCH ROGUES GALLERY

Here we will detail how to run one of the Intel OneAPI samples as a guide for how to use the Intel FPGAs on the CRNCH Rogues Gallery.

We recommend selecting one of the official OneAPI DirectProgramming examples from [this repo](https://github.com/oneapi-src/oneAPI-samples/tree/master/DirectProgramming/C%2B%2BSYCL_FPGA/Tutorials/GettingStarted/fpga_compile). This workflow roughly follows the `fpga_compile` example [<https://github.com/oneapi-src/oneAPI-samples/tree/master/DirectProgramming/C%2B%2BSYCL\\_FPGA/Tutorials/GettingStarted/fpga\\_compile>](https://github.com/oneapi-src/oneAPI-samples/tree/master/DirectProgramming/C%2B%2BSYCL_FPGA/Tutorials/GettingStarted/fpga_compile) from the code samples.

### 38.1 General Workflow

- 1) Evaluate for functional correctness: Build your design for CPU and then for FPGA emulation
- 2) Check mapping to your target board: Look at the optimization report to see if your design fits.
- 3) Build a bitstream
- 4) Run the bitstream on your target board.
- 5) Profile your bitstream.

#### 38.1.1 Which machines can support this flow?

Server Name	Intel FPGA model	FPGA Accelerator Platform Name
flubber2	Arria 10, Stratix 10 (S10)	Bittware 385, Bittware 520N
flubber3	Stratix 10 HBM, S10	Bittware 520N-MX, Intel S10 PAC
flubber9	Agilex	Bittware IA-840F

---

### 38.2 FPGA Emulation

First we request a “development” node to run CPU-based execution and simulation. Note that we can use OneAPI tools across different VMs and servers, so we do some basic testing and debugging using a VM with the appropriate tools.

```
//Request a node from the rg-fpga-dev partition which has all the VMs
salloc -p rg-fpga-dev --nodes=1 --ntasks-per-node=2 --time=01:00:00
salloc: Pending job allocation 67109553
...
```

(continues on next page)

(continued from previous page)

```
salloc: Granted job allocation 67109553
salloc: Waiting for resource configuration
salloc: Nodes rg-fpga-dev1 are ready for job
gburdell@rg-fpga-dev1:~$
```

Then source the Intel OneAPI tools to compile your code. Here we are using the latest tools, but you should also be able to use the 2022.2 toolset.

```
. /tools/intel/oneapi/2023.0/setvars.sh

:: initializing oneAPI environment ...
  bash: BASH_VERSION = 4.4.20(1)-release
  args: Using "$@" for setvars.sh arguments:
:: advisor -- latest
....
:: vtune -- latest
:: oneAPI environment initialized ::
```

Checkout the sample codes and jump to the loop unrolling example. We suggest using your USERSCRATCH directory since it is faster NVMe storage.

```
$> cd USERSCRATCH
USERSCRATCH$> git clone https://github.com/oneapi-src/oneAPI-samples.git
//Go to the top-level of the FPGA tutorial codes
$> cd oneAPI-samples/DirectProgramming/C++SYCL_FPGA/
//Go to the directory for the loop unrolling tutorial
$> cd Tutorials/Features/loop_unroll/
```

**NOTE:** Right now you have to manually edit the main source file to change how selector constructors are defined. This seems to be a bug with newer toolchains.

//Replace the selector\_v; constructors as follows with selector();

```
loop_unroll$ vim src/loop_unroll.cpp
...
#if FPGA_SIMULATOR
  //auto selector = sycl::ext::intel::fpga_simulator_selector_v;
  auto selector = sycl::ext::intel::fpga_simulator_selector();
#elif FPGA_HARDWARE
  //auto selector = sycl::ext::intel::fpga_selector_v;
  auto selector = sycl::ext::intel::fpga_selector();
#else // #if FPGA_EMULATOR
  //auto selector = sycl::ext::intel::fpga_emulator_selector_v;
  auto selector = sycl::ext::intel::fpga_emulator_selector();
#endif
```

You can check the available board support packages using the aoc command. Before doing that, we will want to set the environment to add the Bittware 520N-MX board.

```
#Add the environment for the newer Bittware board
$> . /tools/reconfig/intel/init_env_bittware_pcie.sh

$> aoc -list-boards
```

(continues on next page)



(continued from previous page)

Board list:

```

p520_hpc_m210h_g3x16 (default)
  Board Package: /projects/tools/x86_64/rhel-8/intel-quartus/2020.4/hld/board/
↳ bittware_pcie/s10mx
  Memories:      HBM0, HBM1, HBM2, HBM3, HBM4, HBM5, HBM6, HBM7, HBM8, HBM9, HBM10,
↳ HBM11, HBM12, HBM13, HBM14, HBM15, HBM16, HBM17, HBM18, HBM19, HBM20, HBM21, HBM22,
↳ HBM23, HBM24, HBM25, HBM26, HBM27, HBM28, HBM29, HBM30, HBM31

p520_max_m210h_g3x16
  Board Package: /projects/tools/x86_64/rhel-8/intel-quartus/2020.4/hld/board/
↳ bittware_pcie/s10mx
  Memories:      HBM0, HBM1, HBM2, HBM3, HBM4, HBM5, HBM6, HBM7, HBM8, HBM9, HBM10,
↳ HBM11, HBM12, HBM13, HBM14, HBM15, HBM16, HBM17, HBM18, HBM19, HBM20, HBM21, HBM22,
↳ HBM23, HBM24, HBM25, HBM26, HBM27, HBM28, HBM29, HBM30, HBM31
  Channels:      kernel_input_ch0, kernel_output_ch0, kernel_input_ch1, kernel_output_
↳ ch1, kernel_input_ch2, kernel_output_ch2, kernel_input_ch3, kernel_output_ch3

pac_a10
  Board Package: /net/projects/tools/x86_64/rhel-8/intel-oneapi/2023.0/compiler/2023.
↳ 0.0/linux/lib/oclpga/board/intel_a10gx_pac

pac_s10
  Board Package: /net/projects/tools/x86_64/rhel-8/intel-oneapi/2023.0/compiler/2023.
↳ 0.0/linux/lib/oclpga/board/intel_s10sx_pac

pac_s10_usm
  Board Package: /net/projects/tools/x86_64/rhel-8/intel-oneapi/2023.0/compiler/2023.
↳ 0.0/linux/lib/oclpga/board/intel_s10sx_pac

```

Then use cmake and make to run the FPGA simulation step targeting the S10 board

```

@rg-fpga-dev1:loop_unroll$ mkdir build && cd build
#You can change the device here to another board to investigate different designs
loop_unroll/build$ cmake .. -DFPGA_DEVICE=p520_hpc_m210h_g3x16
-- The CXX compiler identification is IntelLLVM 2023.0.0
...
-- Configuring the design with the following target: p520_hpc_m210h_g3x16
-- Configuring done
-- Generating done
-- Build files have been written to: ../oneAPI-samples/DirectProgramming/C++SYCL_FPGA/
↳ Tutorials/Features/loop_unroll/build

```

And compile the emulation target with make

```

loop_unroll/build$ make fpga_emu
[ 50%] Building CXX object src/CMakeFiles/loop_unroll.fpga_emu.dir/loop_unroll.cpp.o
... Lots of SYCL deprecation warnings you can ignore...
[100%] Linking CXX executable ../loop_unroll.fpga_emu
[100%] Built target loop_unroll.fpga_emu
[100%] Built target fpga_emu

```

You can then run this emulated application and check its output.

```
$> ./loop_unroll.fpga_emu
Input Array Size: 67108864
Running on device: Intel(R) FPGA Emulation Device
unroll_factor 1 kernel time : 321.431 ms
Throughput for kernel with unroll_factor 1: 0.209 GFlops
Running on device: Intel(R) FPGA Emulation Device
unroll_factor 2 kernel time : 313.827 ms
Throughput for kernel with unroll_factor 2: 0.214 GFlops
Running on device: Intel(R) FPGA Emulation Device
unroll_factor 4 kernel time : 87.778 ms
Throughput for kernel with unroll_factor 4: 0.765 GFlops
Running on device: Intel(R) FPGA Emulation Device
unroll_factor 8 kernel time : 80.566 ms
Throughput for kernel with unroll_factor 8: 0.833 GFlops
Running on device: Intel(R) FPGA Emulation Device
unroll_factor 16 kernel time : 85.225 ms
Throughput for kernel with unroll_factor 16: 0.787 GFlops
PASSED: The results are correct
```

## 38.3 Simulation with ModelSim and Questa

Right now we don't support this step.

## 38.4 Investigating the Optimization Report

As with the emulation step, you can generate an optimization report which provides useful statistics about your design for a particular board. This is run as follows:

```
loop_unroll/build$ make report
[ 50%] Building CXX object src/CMakeFiles/loop_unroll_report.a.dir/loop_unroll.cpp.o
[100%] Linking CXX executable ../loop_unroll_report.a
[100%] Built target loop_unroll_report.a
[100%] Built target report
```

To look at your report on the CRNCH servers, it is probably easiest to use the Open OnDemand session. To do this, you can:

- 1) Log into the Open OnDemand instance at [rg-ood.crnch.gatech.edu](http://rg-ood.crnch.gatech.edu) using the instructions [shared here](#).
- 2) Click on the Reconfig tab and select "Virtual Desktop".
- 3) Hit "Launch" on the next page and wait for your VNC session to start.
- 4) Then use the GUI to navigate to your report (it will be located at `loop_unroll/build/loop_unroll_report.prj/reports/report.html`).

## 38.5 Building an FPGA Bitstream

Request an allocation on a larger server node with an appropriate amount of memory.

Here we are requesting 12 cores on flubber 3, 64 GB of RAM, and a time limit of 8 hours.

```
salloc -p rg-intel-fpga-hw --nodes=1 --ntasks-per-node=12 --mem=64G --odelist flubber3 -
↪-time=08:00:00
salloc: Pending job allocation 67109590
salloc: job 67109590 queued and waiting for resources
salloc: job 67109590 has been allocated resources
salloc: Granted job allocation 67109590
salloc: Nodes flubber3 are ready for job
flubber3:~$
```

Switch to your working directory and make sure you've sourced the Intel OneAPI tools and the Bittware environment variables:

```
flubber3$> . /tools/intel/oneapi/2023.0/setvars.sh# Source environment for the Bittware_
↪520N-MX board
flubber3$> . /tools/reconfig/intel/init_env_bittware_pcie.sh
```

Then run `make fpga`. Note this step will likely take a long time to complete - up to 1-2 hours for smaller designs.

```
flubber3$> make fpga
[100%] Linking CXX executable ../loop_unroll.fpga
warning: -reuse-exe file '../oneAPI-samples/DirectProgramming/C++SYCL_FPGA/Tutorials/
↪Features/loop_unroll/build/loop_unroll.fpga' not found; ignored
aoc: Compiling for FPGA. This process may take several hours to complete. Prior to_
↪performing this compile, be sure to check the reports to ensure the design will meet_
↪your performance targets. If the reports indicate performance targets are not being_
↪met, code edits may be required. Please refer to the oneAPI FPGA Optimization Guide_
↪for information on performance tuning applications for FPGAs.
[100%] Built target loop_unroll.fpga
[100%] Built target fpga
```

You then should be able to run your code on the FPGA itself:

```
./loop_unroll.fpga
Input Array Size: 67108864
```

## 38.6 Profiling on FPGA with vTune

This is a more advanced topic that we will add at a later date.



## FPGA POWER MEASUREMENT

One of the big benefits of using an FPGA to accelerate computation is the potential power savings that can be realized over another accelerator like a GPU. However, it's important to be able to measure the power usage of your device while executing a workload.

### 39.1 Intel Power Measurement

Intel provides a few tools to predict power usage during the compilation flow including the [Early Power Estimator](#) (Arria 10 only) and the newer [Intel Power and Thermal Calculator](#) (Stratix 10 and Agilex boards). The newer tools are integrated into Quartus Prime and can be used to estimate power usage before place and route and once a design is fully routed for the platform.

Additionally, some devices like the Programmable Accelerator Cards (PAC) report power and thermal information using their Bus Management Controller interfaces. As shown in the example below, the `fpgainfo power` can be used to sample the current and voltage values for various on-board components.

According to [the BMC guide for Stratix 10](#), this sampling uses the PCIe SMBus to transfer data using a slave I2C interface at a rate from 10 KHz to 1 MHz. This compares reasonably well with other external measurement solutions.

#### 39.1.1 Output from D5005 Stratix 10 PAC card

```
.. example-code::
.. code-block:: JSON
[]$ fpgainfo power
Board Management Controller, microcontroller FW version 2.0.12, RTL version 2.0.6
//***** POWER *****/
Object Id                : 0xEC000000
PCIe s:b:d:f             : 0000:89:00:0
Device Id                : 0x0B2B
Ports Num                : 01
Bitstream Id             : 0x202000200000237
Bitstream Version        : 2.0.2
Pr Interface Id          : 9346116d-a52d-5ca8-b06a-a9a389ef7c8d
MAC address              : 64:4c:36:11:cc:f8
( 2) 12v Backplane Current : 3.02 Amps
( 3) 12v Backplane Voltage : 12.27 Volts
( 6) 1.8v Voltage         : 1.80 Volts
( 7) 1.8v Current         : 8.06 Amps
( 8) 3.3v Voltage         : 3.30 Volts
```

(continues on next page)

(continued from previous page)

```

( 9) 3.3v Current           : 2.37 Amps
(10) FPGA Core Voltage     : 0.88 Volts
(11) FPGA Core Current     : 28.20 Amps
(18) VCCR Voltage          : 1.12 Volts
(19) VCCT Voltage          : 1.12 Volts
(20) VCCR Current          : 7.54 Amps
(21) VCCT Current          : 2.50 Amps
(24) 12v AUX Current       : 2.46 Amps
(25) 12v AUX Voltage       : 12.31 Volts
(30) VCCERAM Current       : 3.06 Amps
(31) VCCERAM Voltage       : 0.90 Volts

[]$ fpgainfo temp
Board Management Controller, microcontroller FW version 2.0.12, RTL version 2.0.6
//***** TEMP *****/
Object Id                  : 0xEC000000
PCIe s:b:d:f               : 0000:89:00:0
Device Id                  : 0x0B2B
Ports Num                  : 01
Bitstream Id               : 0x2020002000000237
Bitstream Version          : 2.0.2
Pr Interface Id            : 9346116d-a52d-5ca8-b06a-a9a389ef7c8d
MAC address                 : 64:4c:36:11:cc:f8
(12) FPGA Core Temperature : 83.50 °C
(26) 12v AUX Temperature   : 32.00 °C
(27) 12v Backplane Temperature : 41.00 °C
(28) 3.3v Temperature      : 41.50 °C
(29) 1.8v Temperature      : 42.00 °C
(32) VCCERAM Temperature   : 48.50 °C
(33) VCCR Temperature      : 59.50 °C
(34) FPGA Transceiver Temperature : 71.50 °C
(35) Board Inlet Air Temperature : 31.00 °C
(36) Board Exhaust Air Temperature : 39.00 °C
(39) RDIMM0 Temperature    : 44.00 °C
(40) RDIMM1 Temperature    : 38.50 °C
(41) RDIMM2 Temperature    : 42.50 °C
(42) RDIMM3 Temperature    : 37.50 °C
(43) VCCT Temperature      : 59.00 °C

```

### 39.1.2 Output from Arria 10 PAC card

```

[]$ fpgainfo power
Board Management Controller, microcontroller FW version 26895
Last Power Down Cause: POK_CORE
Last Reset Cause: None
//***** POWER *****/
Object Id                  : 0xEB000000
PCIe s:b:d:f               : 0000:3D:00:0
Device Id                  : 0x09C4
Socket Id                  : 0x00

```

(continues on next page)

(continued from previous page)

```

Ports Num           : 01
Bitstream Id        : 0x124000200000367
Bitstream Version    : 1.2.4
Pr Interface Id      : 38d782e3-b612-5343-b934-2433e348ac4c
Boot Page           : user
( 0) Total Input Power : 22.00 Watts
( 1) PCIe 12V Current  : 1.77 Amps
( 2) PCIe 12V Voltage  : 12.00 Volts
( 3) 1.2V Voltage      : 1.22 Volts
( 4) 1.2V Current      : 2.66 Amps
( 5) 1.8V Voltage      : 1.81 Volts
( 6) 1.8V Current      : 2.54 Amps
( 7) 3.3V Mgmt Voltage : 3.31 Volts
( 8) 3.3V Current      : 0.72 Amps
( 9) FPGA Core Voltage : 0.90 Volts
(10) FPGA Core Current : 7.29 Amps
(13) QSFP P3V3         : No reading (reading state unavailable)
(16) Core Supply Temp Input : 0.54 Volts
(17) VCCR Voltage      : 1.04 Volts
(18) VCCT Voltage      : 1.04 Volts
(19) VCCR Current      : 1.18 Amps
(20) VCCT Current      : 0.12 Amps
(21) VPP Voltage       : 2.56 Volts
(22) VTT Voltage       : 0.60 Volts

```

### 39.1.3 Intel Power and Thermal Resources

- [Answer Note 787 S10 Thermal Modeling and Management with EPE](#)
- [Stratix10 D5005 Data Sheet](#) - discusses the features of this board and the included BMC, used to gather power and thermal information
- [Stratix10 BMC User Guide](#) - details general BMC options as well as low-level flags used to gather power and other information. Most of this can be accessed using `fpgainfo <command>`
- [Stratix 10 Power Management Guide](#) - talks about SmartVID as well as options for power gating





## RISC-V HARDWARE

Current RISC-V development boards provide an early place to run RISC-V applications and evaluations with standard operating systems. While most of these systems are not yet “high-performance”, we have made several RISC-V platforms available for testing and benchmarking as prototypes.

You also have the option to test RISC-V platforms with [QEMU](#), an open-source emulation and VM tool that can run RISC-V operating systems with RISC-V CPU emulation.

### 40.1 Current Status

### 40.2 BUGS / Feature Requests

- Physical devices are currently being set up with our standard load-out.
- The MangoPi device currently does not support shared home due to its networking setup.

### 40.3 System Specifications

Node Name	CPU	Memory (GB)	Network	Platform Name	Notes
johnny-rv5-1	4xU74 cores and 1xS7 core, 1.4 GHz	16 GB DDR4	1 GB Ethernet	SiFive board	Unmatched
johnny-rv5-2	1 core <a href="#">Allwinner D1</a>	1 GB DDR3	Wifi	<a href="#">MangoPi MPi-MQ1PH</a>	<a href="#">MQ-Pro</a> ; No shared nethome/scratch
johnny-rv5-3	2xU74 cores	8 GB LPDDR4	1 GB Ethernet	<a href="#">StarFive (V1)</a>	<a href="#">VisionFive</a> Currently being deployed

## 40.4 Software and Tools

RISC-V images can currently be run with QEMU - we recommend using [standard Ubuntu RISC-V images](#) or [SiFive's Freedom Unleashed SDK](#).

### 40.4.1 ISA Simulators

- [Spike](#)
- [Dromajo](#)

---

CHAPTER  
**FORTYONE**

---

**FROZONE**



## POWER MONITORING

### 42.1 On-board Power Measurement Tools

- [FPGA Power Measurement](#)
- [NVIDIA GPU](#)
- [AMD GPU with ROCm-SMI](#)

For Bluefield cards, we think we should be using [IPMI](#) to measure power (details forthcoming).

### 42.2 External Power Measurement Tools

These tools use a physical measurement device like a BeagleBoard along with a custom current measurement harness to measure the dynamic current of an attached device. Generally this current is then multiplied by the supply voltage to get the power usage of the device. As an example, a Jetson TX2 board might be supplied by a 12 Volt supply input - if we can measure that that input current is 1.32 Amps we can then determine that the current instantaneous power usage is 15.84 Watts.

There are several different devices that have been proposed for measuring power externally shown in the table below along with links to papers for each device. PowerInsight was productized by [Penguin Computing](#) and WattProf has been produced by [RNET](#).

#### 42.2.1 Sampling Frequencies

This specifies how fast a particular device can sample input currents and voltages. While there is also potentially some latency (e.g., data being reported via USB) in reporting this to the host, this indicates how granular the results are for instantaneous power usage. This table provides a rough estimate of current power monitoring solutions we are aware of.

Device	Type	Sampling Frequency (KHz)
<a href="#">PowerMon 2</a>	External	1 per channel with 3 channels
<a href="#">PowerInsight</a>	External	1
<a href="#">WattProf</a>	External	1-10
<a href="#">Intel S10 FPGAs</a>	On-board	10-1000
<a href="#">NVIDIA GPU (using NVML)</a>	On-board	1-3

### 42.2.2 Related Work

- Karamati, et al., An Energy-Efficient Single-Source Shortest Path Algorithm, IPDPS 2018 - uses PowerMon2
- DeBonis, et al., Qualification for PowerInsight Accuracy of Power Measurements, 2013 - compares PowerMon2 and PowerInsight

## CRNCH ROGUES GALLERY TUTORIALS

This page contains a list of all the recent tutorials that the CRNCH Rogues Gallery has helped to host. For most of these, we have posted any presentations on their respective Github pages under the gt-crnch-rg organization.

### 43.1 2022

- [HPEC - Lucata Pathfinder](#)
- [Telluride - Neuromorphic Tools, Techniques, and Hardware](#) - run by Dr. Jennifer Hasler

### 43.2 2021

- [Telluride - Analog Neuromorphic Tools and Techniques](#) - run by Dr. Jennifer Hasler
- [PEARC - Lucata Pathfinder](#)
- [MICRO - RISC-V Vortex GPGPU](#)
- [DAC - RISC-V](#)





## NEAR-MEMORY RESOURCES

### 44.1 Suggested Papers/Reading

- 

### 44.2 Suggested Background for Georgia Tech Undergrads

- A background in computer architecture with an understanding of concepts like cache and cache coherence, DRAM memory, and memory controllers
- Courses like Processor Design (CS 3220) and Computer Architecture.



## GRAPHBLAS RESOURCES

### 45.1 Why Bother using Linear Algebra packages?

So why go through the effort of installing the library, figuring out how to link, and dealing with C/Fortran issues? The libraries are tuned and optimized. Users can focus on their code details by reducing the amount of code to produce/debug, and more options to switch methods if necessary. In the case of matrix multiplication, for example, the time complexity of the best implementation using a triple loop does not compare with the best parallel and vector implementation using said libraries. The performance is better by a factor of about 4-5.

### 45.2 What is BLAS?

Basic Linear Algebra Subprograms (BLAS) is a specification that prescribes a set of low-level routines for performing common linear algebra operations such as vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication. They are the de facto standard low-level routines for linear algebra libraries; Many highly tuned implementations exist for various problems that can be cast into (Atlas, Flame, LAPACK, etc... )

- [Jack Dongarra - Adaptive Linear Solvers](#) - a nice high-level overview of BLAS-style approaches to solving linear systems. (1 hr, ATPESC 2019)
- [BLAS Tutorial](#)
- [SciNet HPC BLAS and LAPACK](#)

### 45.3 Why do we need a BLAS for Graphs?

Graph problems are challenging to program due to irregular access patterns, poor locality, and difficulty in caching and parallelization. Further, contemporary computer architectures are good at processing linear and hierarchical data structures, such as lists, stacks, or trees, and a massive amount of random data access is required, CPU has frequent cache misses, and implementing parallelism is difficult.

Graph algorithms have a high communication-to-computation ratio. This means that standard latency hiding techniques break down, e.g. pre-fetching and branch prediction provide little benefit.

## 45.4 What is GraphBLAS?

GraphBLAS is a graph-oriented version of linear algebra routines.

It is an API specification that defines standard building blocks for graph algorithms in the language of linear algebra. GraphBLAS is built upon the notion that a sparse matrix can be used to represent graphs as either an adjacency matrix or an incidence matrix. The GraphBLAS specification describes how to graph operations (e.g. traversing and transforming graphs) can be efficiently implemented via linear algebraic methods (e.g. matrix multiplication) over different semirings.

It is believed that the state of the art in constructing a large collection of graph algorithms in terms of linear algebraic operations is mature enough to support the emergence of a standard set of primitive building blocks.

A key insight behind this work is that when a graph is represented by a sparse incidence or adjacency matrix, sparse matrix-vector multiplication is a step of breadth-first search. By generalizing the pair of scalar operations involved in the linear algebra computations to define a semiring, we can extend the range of these primitives to support a wide range of parallel graph algorithms.

In the implementation of GraphBLAS, graphs are encoded as sparse adjacency matrices and use vector/matrix operations to express graph algorithms.

## 45.5 How should I get started learning about GraphBLAS?

There are a wide variety of resources [GraphBLAS-Pointers repo](#). The following listed resources (with stars) are likely the best place to start. We suggest the videos followed by Gábor's tutorial.

### 45.5.1 Suggested Videos

- [Graph Analytics: A Foundational Building Block for the Data Analytics World](#), Tim Mattson, Henry Gabb - a short 13 minute video introducing GraphBLAS
- [GraphBLAS: A linear algebraic approach for high-performance graph algorithms](#)

### 45.5.2 Tutorials

- [Introduction to GraphBLAS: A linear algebraic approach for concise, portable, and high-performance graph algorithms](#) - this is just the slides for the longer version of the above video talk. by Gábor Szárnyas
- [HPEC 2021 Hand-on Tutorial Using Docker](#) - this tutorial allows you to run on your laptop and follow along.

### 45.5.3 Other Suggested Papers/Talks

- [GraphBLAS Wikipedia page](#)
- [Lucata GraphBLAS introduction](#)

## 45.6 What should I do now that I know more about GraphBLAS?

We suggest that you use this new GraphBLAS expertise with the Lucata Pathfinder system. Please see this page for specific information on running GraphBLAS with the Pathfinder.



## NEUROMORPHIC COMPUTING RESOURCES

### 46.1 What do you need to know as background for neuromorphic research?

- **Machine learning concepts**
  - What is a deep neural network?
  - How do methods like stochastic gradient descent work?
- Spiking neural networks - how do they differ from DNNs?

Undergraduate students in this area have suggested the following resources as being helpful for these topics:

- [DeepLizard Backpropagation and Deep Learning Videos](#) - provides an in-depth but approachable introduction to deep learning concepts
- The How to Build a Brain Book by Chris Eliasmith

#### 46.1.1 Suggested Background for Georgia Tech Undergrads

- A background in data structures and machine learning concepts (CS 1332 and 4641)
- Special topics courses like “[Computation and the Brain](#)” (CS8803-CAB) when offered
- Courses from the [neuroscience undergraduate track](#) like “Principles in Neuroscience” (NEURO 2001) would likely also be helpful but may require more of a biomedical background.

#### 46.1.2 Additional Background and Related Topics

Dr. Sam Shapero (GTRI researcher working with LCA and related signal processing topics) suggests understanding the following topics for neuromorphic computing research:

- **Porting DNN applications to SNNs**
  - See this paper as an example for Loihi [Massa, et al., An Efficient SNN for Recognizing Gestures...IJCNN2020](#)
- **Non-linear dynamics and their attractors**
  - ([YouTube video by Dr. Liz Bradley](#))
- Basic neuroscience concepts like feed-forward processing, lateral inhibition, etc.
- **Recurrent structures (SNN) to implement systems of non-linear equations**

- [LCA](#) is one example of implementing a Hopfield network
- See papers like this one: [An Overview of Neuromorphic Computing for AI-Enabled HW-Based Hopfield Neural Network](#)
- **Gradient descent applied to spiking circuits**
  - See papers like this one, [Huh, et al., Gradient Descent for Spiking Neural Networks, NeurIPS18](#)
- **Basins of attraction; chaotic attractors**
  - [Blackmore, Nonlinear Dynamics, Chaos and Strange Attractors with Applications, 2016 slide overview](#)

## 46.2 Videos

- [Nengo and the Neural Engineering Framework \(playlist\)](#)
- [Spiking Neural Networks for More Efficient AI Algorithms](#)
- [Building a Computer Like Your Brain](#)
- [Neuromorphic computing with emerging memory devices](#)
- [Realizing the Promise of Spiking Neuromorphic Hardware - Loihi overview](#)

## 46.3 Books

- [How to build a brain by Chris Eliasmith](#) - a book by one of the current leaders in neuromorphic computing research
- [Neuronal Dynamics](#) - an online book by EPFL
- [Principles of Neural Science](#) - well-regarded book that focuses on the neuroscience aspect of brain functions. Note this does not strictly talk about neuromorphic computing but it provides a good background as to how many neuromorphic systems are designed to mimic the brain.

## 46.4 Courses

- [Computation and the Brain, CS8803 at Georgia Tech](#) - a special topics course that covers the basics of neuroscience, relation to deep learning networks, and theoretical approaches to neuromorphic computing
- [Harvard's Fundamentals of Neuroscience EdX](#)

## 46.5 Survey Papers

- [A Survey of Neuromorphic Computing and Neural Networks in Hardware](#) - a very comprehensive overview of neuromorphic computing from 2017
- [Towards spike-based machine intelligence with neuromorphic computing](#) - not a survey but a nice Nature article overview of neuromorphic computing and related challenges
- [Spiking Neural Networks Hardware Implementations and Challenges: A Survey](#) - a hardware focused survey from 2019



## 46.6 Reports on Neuromorphic Trends

These reports published in the US provide insight into the state-of-the-art and perspective on challenges and future research areas for neuromorphic computing.

1. [Neuromorphic Computing – Architectures, Models, and Applications Workshop, Report](#)
2. [Neuromorphic Computing – From Materials Research to Systems Architecture Roundtable, Report](#)
3. [A Federal Vision for Future Computing: A Nanotechnology-Inspired Grand Challenge](#)
4. [Neuro-Inspired Computational Elements Workshop Report](#)

## 46.7 Other Resources

- [The Computer and the Brain](#) by John von Neumann - an older book from one of the pioneers of computing. Note that neuroscience and computing have both evolved since this was written in 1958!



## QUANTUM COMPUTING RESOURCES

*Last updated: May 4, 2022*

While we don't currently host any quantum hardware, this page lists resources for common simulators, tutorials, and benchmarks that can be used with simulators or cloud-based quantum systems.

### 47.1 What fundamentals do you need to get started with quantum computing research?

- You need a good base knowledge of [linear algebra](#) and some exposure to [quantum mechanics](#) to get started with quantum computing.
  - University of Waterloo has a [nice PDF](#) by Martin Laforest called “The Mathematics of Quantum Mechanics” for their QCSYS summer school that can be used as a high-level introduction to what you need to know.
- Understanding [Dirac notation](#) (also called bra-ket) would also be helpful! [Another resource](#)
- Ideally you also need some initial experience with programming in a language like Python or C/C++. Most quantum computing APIs use Python as their primary language, but some tools like [XACC/QCOR](#) use C++ for most of their software infrastructure.

### 47.2 What related courses does Georgia Tech offer?

- [Linear algebra \(MATH 1554\)](#)
- [Quantum Computing Technologies \(PHYS 4813\)](#) - Dr. Colin Parker
- [Quantum Information and Quantum Computing \(PHYS/MATH 4782\)](#)
- [Introduction to Quantum Computing \(CS8803/ECE8803\)](#) - Dr. Moin Qureshi
- [Quantum computing devices and Hardware \(CS8863\)](#) - Dr. Asif Khan

## 47.3 What are the “best” resources to get started with?

Dr. Eugene Dumitrescu at ORNL has suggested the following resources, which include an accurate mathematical picture of quantum computing.

- [Nielsen and Chuang’s Quantum Computation and Quantum Information](#) - Sometimes referred to as Mike and Ike, this book is still very relevant even though it was first published in 2012! Most people consider this the most important book on the topic.
- [Quantum Computing: A Gentle Introduction](#) - recommended by Prof. Moin Qureshi in addition to Nielsen and Chuang and has some added info on quantum algorithms
- [Griffith’s Quantum Mechanics book](#)
- [John Preskill’s notes for Physics 219](#)
- [Scott Aaronson’s notes for his Intro to Quantum Information Science course](#)
- [Video lectures from CMU](#)
- [MIT OpenCourseWare \(OCW\)](#)

### 47.3.1 Vendor-provided Tutorials

IBM’s [Circuit Composer](#) and [Qiskit Notebooks](#)

- [Qiskit textbook](#)
- [Qiskit for Educators](#)
- [Hello World with Qiskit](#) - not designed by IBM

[Microsoft’s Quantum Katas using Q#](#) - Uses Jupyter notebooks to introduce simple quantum concepts where each Kata investigates one technique or idea.

[DWave Tutorials](#) (quantum annealing)

### 47.3.2 University Tutorials

[EPiQC Tutorial on Quantum Computing](#): Includes a Docker image to run all the simulation tools on your local machine. Based in part on Scaffold compiler tools and Qiskit.

[NC State Quantum Primer for IBM Q and DWAVE](#)

### 47.3.3 Quantum Algorithms and Survey Papers

[Next Steps in Quantum Computing: Computer Science’s Role](#)

[QDB: From Quantum Algorithms Towards Correct Quantum Programs](#) - this paper from Princeton describes some quantum algorithm/benchmark implementations using ScaffCC and Qiskit.

[Overview and Comparison of Gate Level Quantum Software Platforms](#) - July 2018 paper

## 47.3.4 Software

## 47.4 Qiskit

- [Qiskit installation](#)

## 47.5 QCOR (ORNL)

- [Get started quickly with XACC using Docker](#)
- [QCOR Examples](#)

## 47.6 XACC (ORNL)

- [Get started quickly with XACC using Docker](#)
- [XACC Tutorial](#)
- [XACC Python Examples](#)

### 47.6.1 Quantum Hardware Testbeds and Resources

- [IBM Q](#)
- [Microsoft Azure Quantum](#)
- [Rigetti Quantum Cloud Service and Forest SDK](#)
- [DWave Leap \(quantum annealing\)](#)

While the above resources are considered primary resources, the following links are provided as additional resources.

### 47.6.2 Introductory videos about quantum computing

These videos might provide a good high-level introduction to quantum computing and were put together from the IEEE Quantum Computing Education Effort (to be publicly announced).

- [Talia Gershon of IBM Explains Quantum Computing in 5 Levels of Difficulty \(20 min\)](#)
- [Quantum Computing - Top 3 Microsoft Breakthroughs with Krysta Svore \(25 min\)](#)
- [Quantum Computers Explained; Limits of Human Technology by nova.org.au \(7 min\)](#)
- [John Preskill's Keynote Lecture at Q2B on 5 Dec 2017 \(50 min\)](#)
- [Quantum Computing for Dummies : A Simple Explanation for Normal People by Sean Ong \(6 min\)](#)
- [Quantum Computer in a Nutshell produced by Pawel Dobosz \(30 min\)](#)
- [Quantum theory: It's Unreal by Terry Rudolph \(1 hr\)](#)
- [Quantum Computers Animated by John Preskill & Spiros Michalakis \(7 min\)](#)
- [You Don't Know How Quantum Computers Work! by Frame of Essence \(15 min\)](#)
- [The Mathematics of Quantum Computers | Infinite Series \(12 min\)](#)

### 47.6.3 Other books and Resources

[Programming Quantum Computers \(book\)](#) - available via O'Reilley digital library

[Quantum Tutorial for Architects](#)

[Brilliant.org Quantum Computing Course](#)

[From Cbits to Qbits](#) - David Mermin paper on teaching students without a background in quantum physics.

[Dancing with Qubits \(book\)](#) - Suggested for K-12 and undergrads as it includes intro material for linear algebra and computational complexity.

### 47.6.4 Educational Resources

[Qureca site](#) - links to other online quantum computing educational resources

[Qutools](#) - German site focused on teaching quantum physics

[FutureLearn MOOC](#) - a five week MOOC focused at high-level quantum learning

### 47.6.5 Other QC Software

## 47.7 ScaffCC / Scaffold

- [ScaffCC Github](#)

## 47.8 Quipper

- [Quipper: A scalable quantum programming language](#) - a functional language for quantum simulation

## 47.9 CIRQ/Open Fermion (Google Research)

- [CIRQ](#) - Python library for simulating quantum circuits
- [Open Fermion](#) - Library for simulating fermionic systems

## RELATED TESTBED RESOURCES

*Last updated: September 24th, 2021*

We realize that the Rogues Gallery cannot host all possible hardware, software, and configurations, so we encourage researchers to check out related resources for their work. Note that this page is provided for informational purposes, but we at the Rogues Gallery cannot provide any additional information or support for users of these other testbeds.

### 48.1 Reconfigurable Computing

Testbed	Sponsor	Who Can Apply	Notes
<a href="#">Xilinx Adaptive Compute Cluster</a>	Xilinx University Program	Non-commercial researchers	Hosted at UIUC and ETH
<a href="#">Intel DevCloud</a>	Intel	Anyone	Stratix 10 and Arria10; 3 month time limit
<a href="#">Open Cloud Testbed</a>	NSF CCRI #1925464	US researchers	Alveo U280; networking; virtualization
<a href="#">ACES - Texas A&amp;M</a>	NSF #2112356	TBD	2022 deployment

### 48.2 Related Tutorials

- [Open Cloud Testbed Tutorial](#)