Intel FPGA Devcloud Workshop

1. Introduction

The Intel FPGA Devcloud offers users access to Xeon servers connected to the latest Intel FPGA Platform Acceleration Cards (PAC). In this lab you will learn how to connect to the Intel FPGA Devcloud, and run the design flow for OpenCL workloads. The Intel FPGA Devcloud also offers compilation for dpc++ workloads as well as RTL Acceleration Functional Units (AFU). This is not a language or design course; this course strictly focuses on access to the Intel FPGA Devcloud and runs through interactive and batch mode compilation of a Quickstart exercise.

There are numerous methods to connect to the Intel FPGA Devcloud including using putty, Mobaxterm and JupyterLab.. Multi window graphics are enabled through X2Go. We will focus on Mobaxterm, with some mention of other techniques. The account you create will expire in 90 days. Should you want to continue use, pay attention to a series of expiration emails that alert you to account expiration and follow the renewal instructions.

2. Flow Documentation for the Intel FPGA Devcloud

The Intel FPGA Devcloud support team maintains a github site with all the connectivity and tool access methods required for productive operation of our free cloud service. This lab manual is extracted directly from our Devcloud access instructions at this link. The github instructions cover many topics. This abbreviated version will focus on getting you started using the devcloud. The github site covers several areas you might want to cover on your own once completing this workshop.

3. Account Access

These instructions detail how to connect your Windows PC to the devcloud which runs Linux. If you are attempting connectivity from a Mac or Linux machine, we do not offer exact instructions, but you can use the ssh key and config file details described here with other ssh terminals. Today's lab assumes you are on a PC and will connect via Mobaxterm.

The devcloud has 3 instances: FPGA, OneAPI and Edge. FPGA accounts offer you a superset of FPGA devcloud and OneAPI devcloud features. If you have a devcloud account and can login into the LINUX environment, you can check if you are on the FPGA Devcloud instance which is required for todays lab. If you are not, follow the sign up procedure below. Your setup will support OneAPI, OpenCL and RTL AFU

compilations. For general digital logic type projects using lower complexity boards (not heterogeneous computing), you would be better off acquiring a board and using our Quartus Prime Lite free download software.

Once logged in, type groups. If you see the entry c009-fpga, you have access to the FPGA devcloud instance and you can skip the next section. If you don't, go through the account access steps. If you completed the earlier lab today on OneAPI, you can use the same credentials you used to access that JupyterLab based workshop.

With the FPGA instance you can run all of the features of the ONEAPI instance and gain access to an additional 12 machines that have FPGA PAC cards and associated tools for OpenCL/RTL AFU workloads.

Note: the default account duration is 90 days. You will get reminders to renew your account if you need access for more than 90 days. If you allow your account to expire, you will lose the data in your account. We recommend copying source files back to your local machine for safe keeping.

If you already have an Devcloud account, you may skip to the next step.

Please use this cloud website landing page to submit a request to access the FPGA Cloud:

https://intelsoftwaresites.secure.force.com/fpgadevcloud

Once you've signed up, you should get an immediate screen response with your new user ID and instructions on how to set up your account. You will also receive a follow up email from Intel Devcloud which should take <5 minutes giving you a record of your user ID and your user ID key for login. This is an example of the resulting email which will be sent to you:

```
Welcome "user name",

We are excited that you chose Intel® FPGA Cloud.  Free access. No downloads. No
installations. No maintenance.

Your account should already be activated. Below are your credentials for your
reference.
Unique Access URL: https://devcloud.intel.com/fpga/?uuid=cd1d...

User ID: u12345
UUID Key: cd1d...

Access to Intel® FPGA DevCloud typically expires after 120 days. However, longer
access times are granted based on user request. To extend this access, please
```

```
email fpgauniversity@intel.com and give the extension time needed to complete your
project.

Getting started with FPGA Cloud:
1. Access detailed instructions for environment setup: https://github.com/intel/FPGA-
Devcloud/tree/master/main/Devcloud_Access_Instructions#devcloud-access-instructions
2. If you have technical questions or recommendations, please post them to our FPGA
forum: https://forums.intel.com/s/topic/0TO0P000000MWKFWA4/application-acceleration-
with-fpgas
3. If you have unresolved issues, email fpgauniversity@intel.com and give us a
detailed description of your problem.
It's all about you and your code. We look forward to the innovations you'll create.

- Your friendly Intel DevCloud Team
```

Once you have an account / email received you are ready to start the process to setup your account within the cloud.

There are different methods of terminal connections. Listed below are a few options you can select in choosing which Terminal application tool you would like to use. This guide focuses on the recommended method 1.

- Windows with Mobaxterm (SSH client) (recommended)
- Windows with Cygwin
- Windows with PuTTy
- Access through Jupyter Notebooks

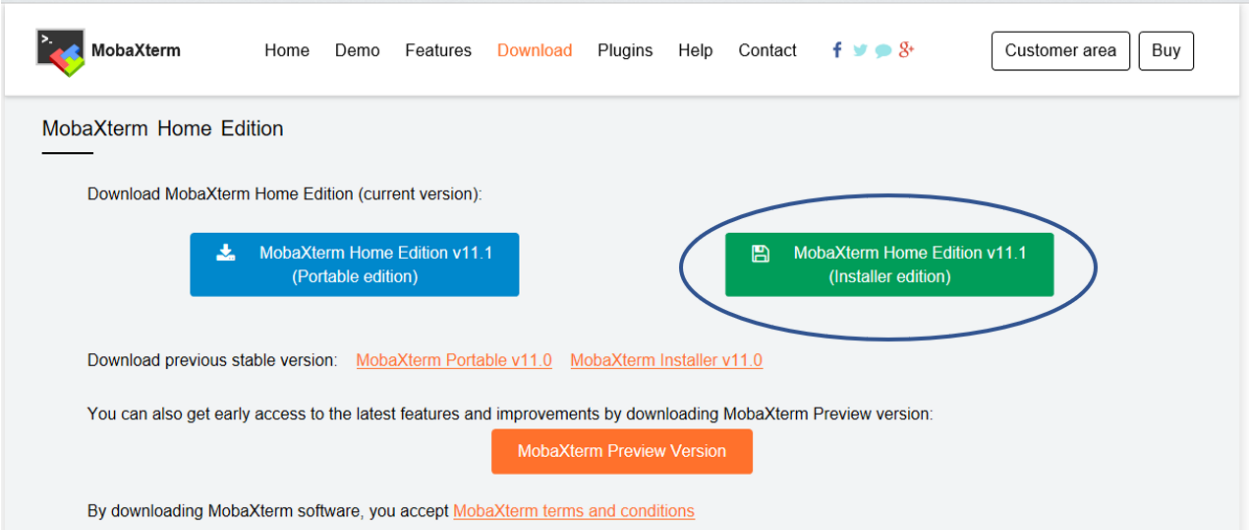
4. Access from your PC via MobaXterm or from Linux Terminal

MobaXterm is an enhanced terminal for Windows with an X11 server, a tabbed SSH client and several other network tools for remote computing (VNC, RDP, telnet, rlogin). **MobaXterm** brings all the essential Unix commands to Windows desktop, in a single portable exe file which works out of the box and makes your Windows PC look like a UNIX environment. Note that the X11 server with Mobaxterm runs slow, so we don't recommend its use. The X2Go section shows how to gain GUI access.

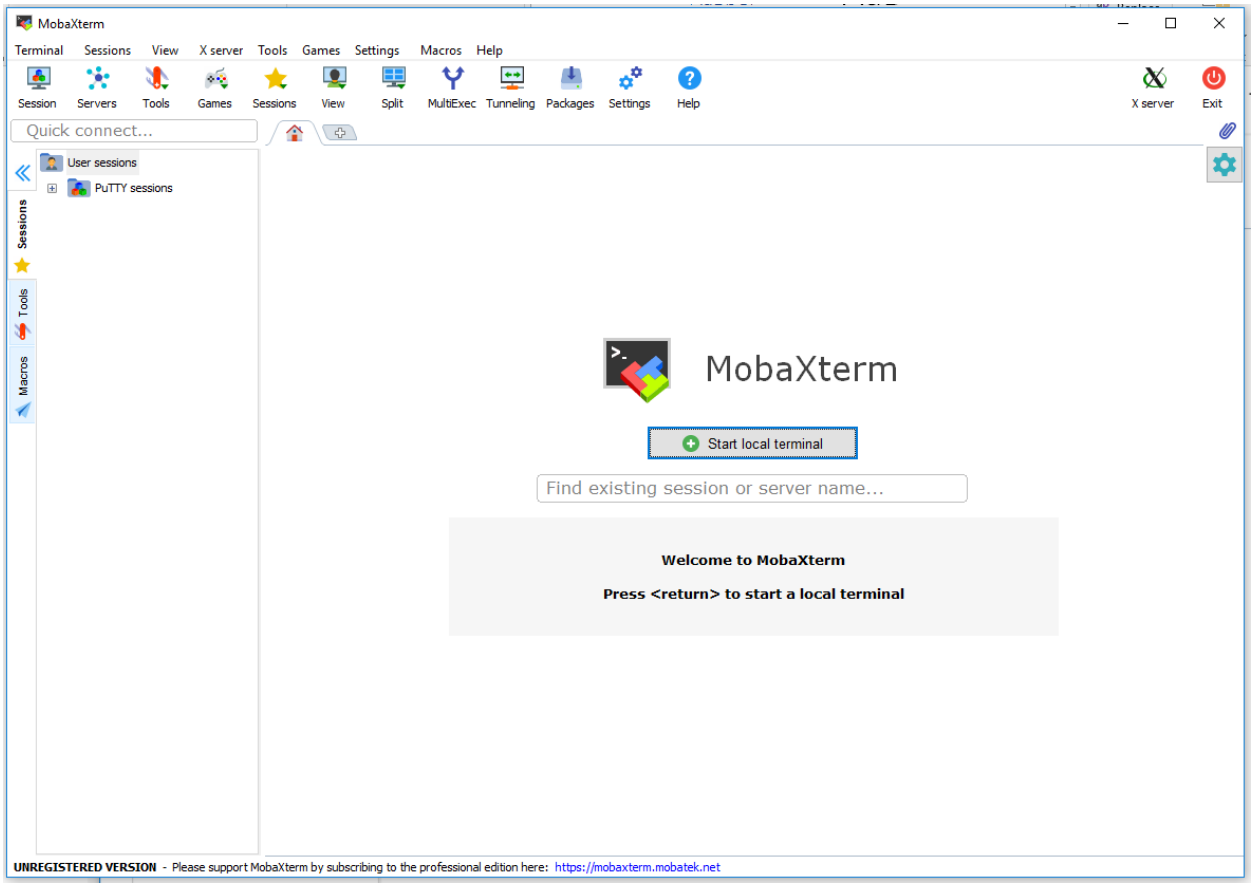
Install MobaXterm

- Download the MobaXterm free edition:

Note: Get the **installer edition**, not the portable edition. (The installer edition will enable you to save login profiles.) . Download the zipfile, first extract it and when complete, click on the msi file to install Mobaxterm.

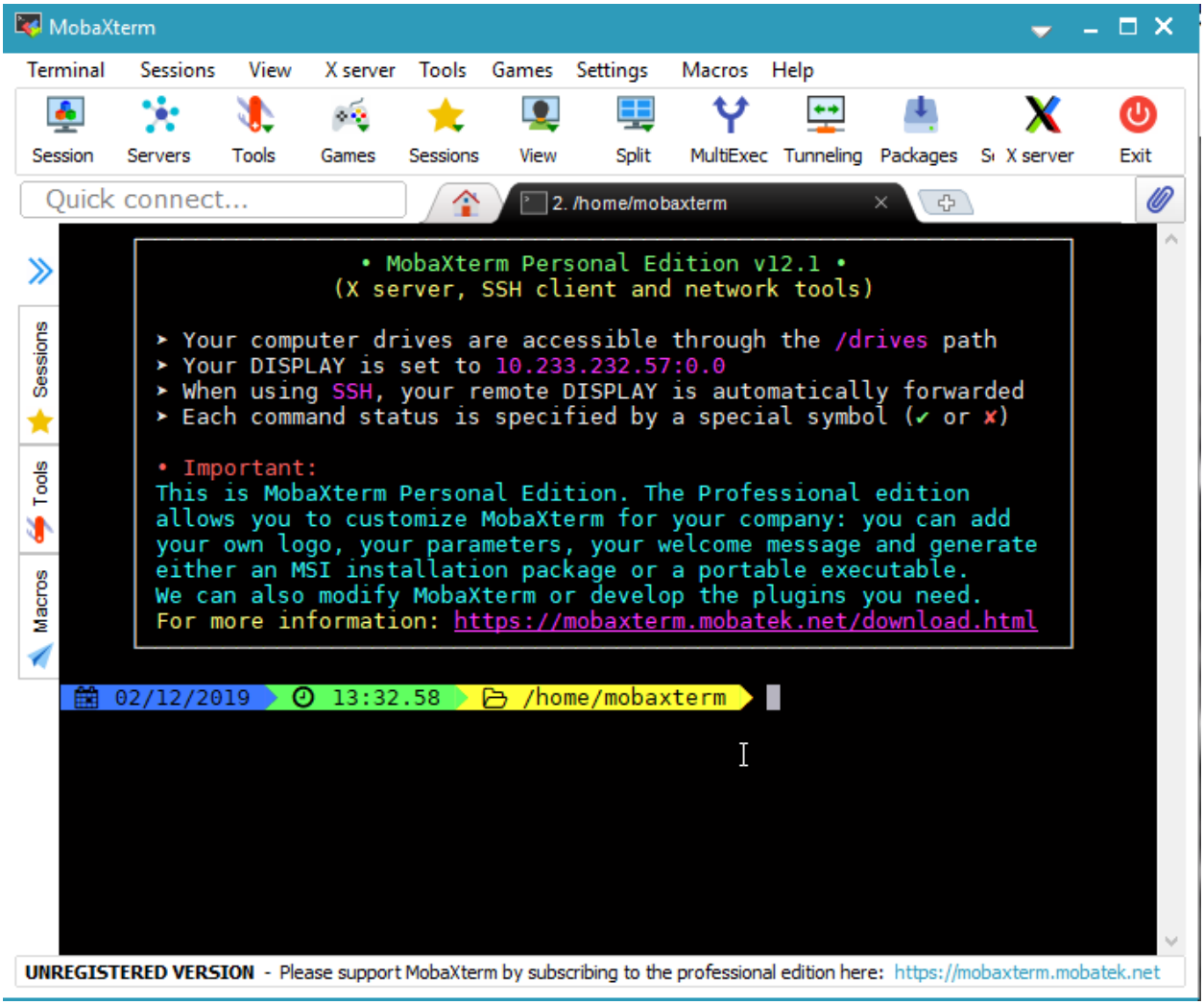


- After the installer completes, launch MobaXterm using the installer. You should see the following:



- Click: **"Start local terminal"**. Within this console you can see your local PC based files using standard Linux operating system commands (ls, cd, vi and etc.).

- Navigate around with `cd` (change directory) and `ls` (list) you will recognize your Windows folders and files accessible through a UNIX interface. To view your "C" drive in Windows, cd /drives/c . If you type ls and cd you can navigate the hierarchy that you see in Windows File Explorer. When you navigate to home via the cd command, you are at C:\Users<username>\Documents\MobaXterm\home on your PC.
  Return to home by typing: `cd`

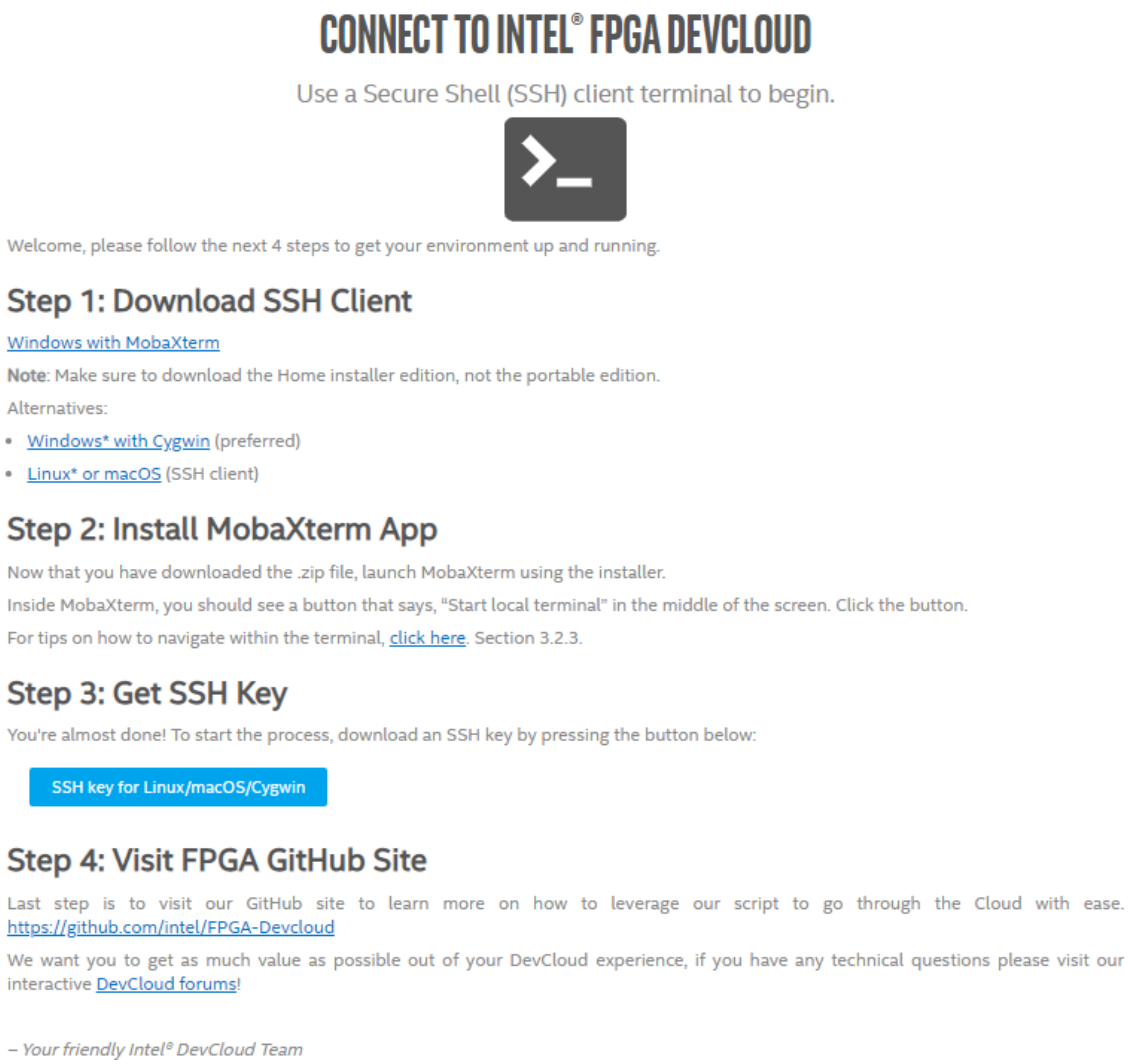


5. Downloading an SSH key to get access to the FPGA devcloud

To start the process:

- Click on the following link to access the Connect website: https://devcloud.intel.com/fpga/connect/#.

- Once you sign in, the following page will then be displayed. Click on the "SSH key for Linux/macOS/Cygwin" blue button under Step 3: Get SSH Key to download your key.



CONNECT TO INTEL® FPGA DEVCLOUD

Use a Secure Shell (SSH) client terminal to begin.

Welcome, please follow the next 4 steps to get your environment up and running.

Step 1: Download SSH Client

Windows with MobaXterm

Note: Make sure to download the Home installer edition, not the portable edition.

Alternatives:

- Windows* with Cygwin (preferred)
- Linux* or macOS (SSH client)

Step 2: Install MobaXterm App

Now that you have downloaded the .zip file, launch MobaXterm using the installer.

Inside MobaXterm, you should see a button that says, "Start local terminal" in the middle of the screen. Click the button.

For tips on how to navigate within the terminal, click here. Section 3.2.3.

Step 3: Get SSH Key

You're almost done! To start the process, download an SSH key by pressing the button below:

SSH key for Linux/macOS/Cygwin

Step 4: Visit FPGA GitHub Site

Last step is to visit our GitHub site to learn more on how to leverage our script to go through the Cloud with ease. https://github.com/intel/FPGA-Devcloud

We want you to get as much value as possible out of your DevCloud experience, if you have any technical questions please visit our interactive DevCloud forums!

– Your friendly Intel® DevCloud Team

- Once you have downloaded the SSH key, return to these Instructions.
- Create the directory ~/. ssh, unless it already exists and move the private SSH key into permanent storage in ~/.ssh:
- mkdir -p ~/.ssh

mv /drives/c/Users/<user>/Downloads/devcloud-access-key-#####.txt ~/.ssh/

- Add the following lines to files ~/.ssh/config. Swap the u# with your own you received in your sign-up email.

host devcloud

```
#replace with your own user name
user u#####
IdentityFile ~/.ssh/devcloud-access-key-#####.txt
ProxyCommand ssh -T -i ~/.ssh/devcloud-access-key-#####.txt guest@devcloud.intel.com
```

If you saved your key in a location other than /drives/c/Users//Downloads, insert the correct path and the correct user number that was provided to you in the email.

- Set the correct restrictive permissions on the private SSH. Run the following commands in terminal:

- chmod 600 ~/.ssh/devcloud-access-key-u#####.txt
  chmod 600 ~/.ssh/config

- Connection to Devcloud

Public User – for use outside Intel Devcloud

After the preparation steps above, you should be able to log in to your login node in the Intel Devcloud without a password. If you are behind the Intel firewall, refer to the github site for a different method.

Enter the following:

```
ssh devcloud
```

Upon the first login, you will be asked to add the host devcloud to the list of known hosts. Answer **yes**.

```
The authenticity of host 'devcloud' (<no hostip for proxy command>)' can't be
established.
ECDSA key fingerprint is SHA256:...
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'devcloud' (ECDSA) to the list of known hosts.
# We are in!
logout
Connection to login-1 closed.
```

Next time you log in, you will only need to type `ssh devcloud`

Confirm you can login to the Intel FPGA devcloud from mobaxterm. You should now be on node login-2. This node is referred to as the head node.

Proceed to step and explore the different options with the devcloud_login and tools_setup commands. Note some of the tool logs for devcloud_login and tools_setup in the github README might differ slightly from what you see as the scripts get updated frequently to utilize new features.

When looking at devcloud_login, don't launch any jobs, just take a look at the features offered. If you do login to a node and exit and try to recapture another node, sometimes it takes 4 minutes to reclaim another compute node (this a weird quirk in the Linux OS we are working to fix). Take a look at option 6 in devcloud_login – this will give you a summary of available resources for various workload compilations. This will give you a map of the computation resources accessible in the devcloud. Arria 10 is 20nm silicon and Stratix 10 is 14nm silicon.

Now we will have you run a few different types of workload compilations to gain some practice. The goal of this section is to demonstrate the flow, not the language constructs.

At this point you should be at login-2 (the head node). Note multiple users can be logged into the headnode, however only a single user can access a compute node. We recommend batch mode processing launched from the head node whenever possible to not consume compute resources with interactive jobs.

6. Connecting to Servers Running FPGA Development Software

- Understanding available resources

You are now logged into machine called login-2 (headnode). You cannot run compute jobs here. You need to run compute jobs on a powerful compute node server.

Nodes can run Quartus, OpenCL or dpc++ emulation and compilation.

Some nodes can connect to machines with the above capabilities and also are directly connected to Arria 10 and Stratix 10 PAC cards.

- Login Script

To facilitate connectivity without needing to understand Linux commands, we offer a script that simplifies connectivity called devcloudLoginToolSetup.sh located under /data/intel_fpga/devcloudLoginToolSetup.sh . Modify your ~/.bashrc by including these three lines inside your ~/.bashrc:

```
if [ -f /data/intel_fpga/devcloudLoginToolSetup.sh ]; then
    source /data/intel_fpga/devcloudLoginToolSetup.sh
fi
```

Source your .bashrc file from the login-2 command prompt.

Now let's test out a few devcloud login commands:

Run the devcloud_login command. The picture below shows an example output. It displays a list of functions available nodes can run, or allows you to connect to a specific known node.

```
u27224@login-2:~$ devcloud_login

You are selecting an interactive compute server sesssion. Please consider using batch mode submission using
devcloud_login -b to not tie up compute servers with idle sessions.
See the help menu using devcloud_login -h for more details.

What are you trying to use the Devcloud for?

1) Arria 10 PAC Compilation and Programming - RTL AFU, OpenCL
2) Arria 10 - OneAPI, OpenVINO
3) Stratix 10 PAC Compilation and Programming - RTL AFU, OpenCL
4) Stratix 10 - OneAPI, OpenVINO
5) Compilation (Command Line) Only
6) Enter Specific Node Number

Number:
```

```
Number: 6
Showing available nodes below: (31 available/63 total)
-------------------------------------------------------------------------------
Nodes with no attached hardware: (11 available/35 total)
s001-n045 s001-n067 s001-n069 s001-n070 s001-n072 s001-n073 s001-n074 s001-n075 s001-n076 s001-n077 s001-n078
-------------------------------------------------------------------------------
Nodes with Arria 10: (5 available/8 total)
Release 1.2:

Release 1.2.1:
s005-n001 s005-n002 s005-n003 s005-n004 s005-n007
-------------------------------------------------------------------------------
Nodes with Arria 10 OneAPI: (8 available/12 total)
s001-n081 s001-n082 s001-n084 s001-n086 s001-n087 s001-n088 s001-n089 s001-n090
-------------------------------------------------------------------------------
Nodes with Stratix 10: (3 available/5 total)
s005-n005 s005-n006 s005-n009
-------------------------------------------------------------------------------
Nodes with Stratix 10 OneAPI: (3 available/3 total)
s001-n142 s001-n143 s001-n144
-------------------------------------------------------------------------------

What node would you like to use?

Node:
```

Once you select a node to start an interactive login, it will also output the command required to set up the x2go graphics window. This isn't required in this section.

Other features to the Devcloud login script are the ability to display available nodes, submit batch jobs from the headnode as well as to speed the user's interaction when wanting to log into a compute node interactively.

For more information, try "devcloud_login --help" on a terminal that is logged into the FPGA devcloud.

```
u12345@login-2:~$ devcloud_login --help

Usage:
------

devcloud_login -h | --help
devcloud_login -l
devcloud_login -I <script args options>
devcloud_login -b <script args options> [walltime=hh:mm:ss] <job.sh>
devcloud_login

Description:
------------

devcloud_login is a command to display available nodes, start an interactive login to a compute
node, or submit a batch job to a compute node.

Argument Options:
-----------------

A10PAC  (eg. devcloud_login -I A10PAC 1.2)         Arria 10 PAC; 1.2  1.2.1
A10OAPI (eg. devcloud_login -I A10OAPI)            Arria 10 OneAPI
S10PAC  (eg. devcloud_login -I S10PAC)             Stratix 10 PAC
CO      (eg. devcloud_login -I CO)                 Compilation Only
SNN     (eg. devcloud_login -I SNN s001-n139)      Specific Node Name

Batch Submissions:
------------------
Walltime is optional; use if batch job needs more than 6 hours. Maximum Walltime is 48 hours.

A10PAC  (eg. devcloud_login -b A10PAC 1.2 [walltime=12:00:00] job.sh)      Arria 10 PAC; 1.2  1.2.1
A10OAPI (eg. devcloud_login -b A10OAPI [walttime=12:00:00] job.sh)         Arria 10 OneAPI
S10PAC  (eg. devcloud_login -b S10PAC [walttime=12:00:00] job.sh)          Stratix 10 PAC
CO      (eg. devcloud_login -b CO [walttime=12:00:00] job.sh)              Compilation Only
SNN     (eg. devcloud_login -b SNN s001-n139 [walttime=12:00:00] job.sh)   Specific Node Name
```

- Development Tool Access and Setup

From a terminal that is logged into a compute node, to gain Quartus Access and Quartus Setup you can source the bash scripts manually however its highly recommended to use the **tools_setup** command. This command will guide you through query of what compile workload you want to run and setup environment variables and search paths.

Another feature to the tools-setup function is the ability to speed the user's interaction when wanting to source the environment variable settings for a tool when interactively logged into a compute node. Instead of answering "Which tool would you like to source? ..." every time, you can type the following:

```
tools_setup -t <argument option>
---------------------------------Argument-Options--------------------------------
QL <Version>
QS <Version>
QP <Version>
HLS <Quartus Edition> <Version>
A10DS
A10OAPI
S10DS
S10OAPI
```

The above command can also be included in a batch script. Simply include the following two lines to be able to use the tools-setup function within your batch script.
Then submit a batch job.

```
source /data/intel_fpga/devcloudLoginToolSetup.sh
tools_setup -t <argument option>
```

For more information, try "tools_setup --help" on a terminal that is logged into the devcloud.

```
u12345@s001-n039:~$ tools_setup --help

Usage:
------

tools_setup -h | --help
tools_setup -t [<script args options>]
tools_setup

Description:
------------

tools_setup is a function aimed to help the user setup an environment variable in a devcloud node.
The tools_setup has a user interactive and non interactive mode.

Argument Options:
-----------------

QL      (eg. tools_setup -t QL 18.1)          Quartus Lite; 18.1
QS      (eg. tools_setup -t QS 18.1)          Quartus Standard; 18.1
QP      (eg. tools_setup -t QP 18.1)          Quartus Pro; 17.1 18.1 19.2 19.3 20.1
HLS     (eg. tools_setup -t HLS QL 18.1)      High-Level Synthesis
A10DS   (eg. tools_setup -t A10DS 1.2)        Arria 10 Development Stack
A10OAPI (eg. tools_setup -t A10OAPI)          Arria 10 One API
S10DS   (eg. tools_setup -t S10DS)            Stratix 10 Development Stack
```

When you are done using the compute node, please logout with Ctrl-F or exit.

8. Quickstart Guides

A series of Quickstart guides are published here: https://github.com/intel/FPGA-Devcloud/tree/master/main/QuickStartGuides . We copy the information from the Arria 10 Quick start guide here.

We will start with running an emulation of an OpenCL workload. We will only run emulation mode and not compile for the FPGA due to time constraints as the FPGA compiles can take up to an hour.

Run this interactively (quickly so as to not tie up the machine for too long). All steps should take 5 minutes or less.

- Initial Setup

Run the devcloud_login command and connect to an Arria 10 capable node. This function is available in the script: /data/intel_fpga/devcloudLoginToolSetup.sh .

Select option 1 or option 6 and connect to an Arria 10 ready compute node.

Note: If no node is available, you can skip to the batch mode section of the manual, or Jupyter Lab section of this manual. We only have a handful of machines, so it's possible that you might get in a queue.

Once on this node, run tools_setup.

```
tools_setup
```

Select the Arria 10 PAC Compilation and Programming - RTL AFU, OpenCL option.

Make working directory

```
mkdir A10_OPENCL_AFU
```

We will then copy the example folder into this project folder.

Use version 1.2.1 and type this into the terminal:

```
cp -r /opt/intelFPGA_pro/quartus_19.2.0b57/hld/examples_aoc/hello_world A10_OPENCL_AFU
cp -r /opt/intelFPGA_pro/quartus_19.2.0b57/hld/examples_aoc/common A10_OPENCL_AFU
cd A10_OPENCL_AFU
```

- Running OpenCL in emulation mode

The first step of the OpenCL flow is to compile and execute the design for emulation mode. This step allows you to quickly verify the functionality of your code on the CPU without performing the conversion from OpenCL to RTL and from RTL to an FPGA executable, which takes up to an hour.

```
cd hello_world
aoc -march=emulator -v device/hello_world.cl -o bin/hello_world_emulation.aocx
ln -sf hello_world_emulation.aocx bin/hello_world.aocx
```

The next step is to compile the host code. Note: use make clean followed by make to force a recompile.

```
make clean; make
```

Now run for the host code binary.
Note that the with the environment setting shown, the host code knows the .aocx file is for emulation execution on the CPU and not on the FPGA card.

For version 1.2.1, run emulation with this command:

```
./bin/host -emulator
```

You should see a list of parameters and Kernel execution is complete.

Log out of compute node using the exit command. You will be back at the head node login-2.

Let's repeat these steps using batch mode processing.

Make a clean directory. Call it BATCH.

mkdir BATCH

cd batch

vi A10_v1.2.1_opencl_batch.sh

Add these commands by copy and paste:

```
# Initial Setup
source /data/intel_fpga/devcloudLoginToolSetup.sh
tools_setup -t A10DS

# Copy Over sample design
cp -r /opt/intelFPGA_pro/quartus_19.2.0b57/hld/examples_aoc/hello_world A10_OPENCL_AFU
```

```
cp -r /opt/intelFPGA_pro/quartus_19.2.0b57/hld/examples_aoc/common A10_OPENCL_AFU
cd A10_OPENCL_AFU

# Running project in Emulation mode
cd hello_world
printf "\\n%s\\n" "Running in Emulation Mode:"
aoc -march=emulator -v device/hello_world.cl -o bin/hello_world_emulation.aocx
# Creating symbolic link to emulation .aocx
ln -sf hello_world_emulation.aocx bin/hello_world.aocx
make
# Run host code for version 1.2.1
./bin/host -emulator
error_check
```

From the headnode login-2, run this command:

```
devcloud_login -b A10PAC 1.2.1 A10_v1.2.1_opencl_batch.sh
```

To see the resulting terminal output, consult the files:

A10_v1.2.1_opencl_batch.sh.exxxxxx
A10_v1.2.1_opencl_batch.sh.oxxxxxx

xxxxxxx is a unique job ID. The .exxxxxx file is the error log and the .oxxxxxx file is the terminal log where success or failure of the commands can be determined.

Note the log files are created at the very end of execution. You can observe status of your batch job with the qstatus command. It should take about 5 minutes to complete this task. If it takes longer it could be because you are in the queue.

You can observe the status of queue with the qstatus command. Queue commands can be killed using the qdel command:

qdel 18225.v-qsvr-fpga.aidevcloud

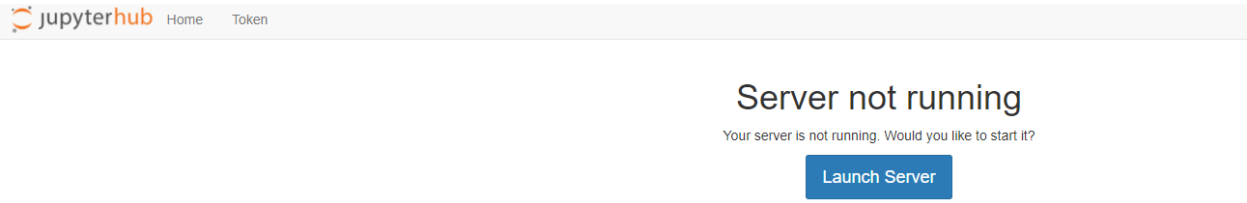
Note that the qstatus command cuts off the entire job ID and will only show 18225.v-qsvr-fpga.aide and leave off the entire aidevcloud part. That is needed for qdel to recognize the Job ID.

9. Using JupyterLab

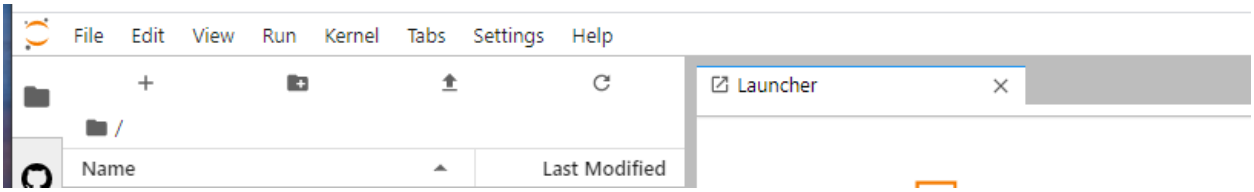
Another means to connect to the Intel FPGA Devcloud is through JupyterLab. If you took the morning OneAPI lab, you will be familiar with this. This bypasses Mobaxterm and allows access through a web interface.

Start by clicking this startup link:
https://jupyter.oneapi.devcloud.intel.com/hub/user/u27224/lab/tree/Welcome.ipynb

Click on the Launch Server Link



You will get a web view of your devcloud account. Click on the + icon for launcher:



Once in launcher select the terminal icon. This will log you in to a OneAPI compute node. To utilize the devcloud_login and tools_setup command flow:

ssh login-2

Continue the flow documented for mobaxterm. Jupyterlab has a myriad of features and multi-window terminal capabilities. Refer to JupyterLab documentation for details.

10. Using Graphics Programs

Should you utilize GUI based programs (for example, Quartus), we utilize X2Go for graphics. Installation and operation is described in detail here.

Refer to the section Graphics usage on the FPGA Devcloud.

11. Summary

This concludes the lab. For further studies, consult the git hub site Quickstart guides for instructions for OneAPI (dpcpp) or RTL AFU workload compilations, increasing walltime for longer compilations and many other features. Remember, although we demonstrate interactive mode and batch mode, you should utilize batch mode launch from login-2 to enable better sharing of resources across users.

Date	Author	Comments
2/26/2021	L. Landis	Initial Release