

My notes on how to follow along with FPGA lab on Wednesday November 3.

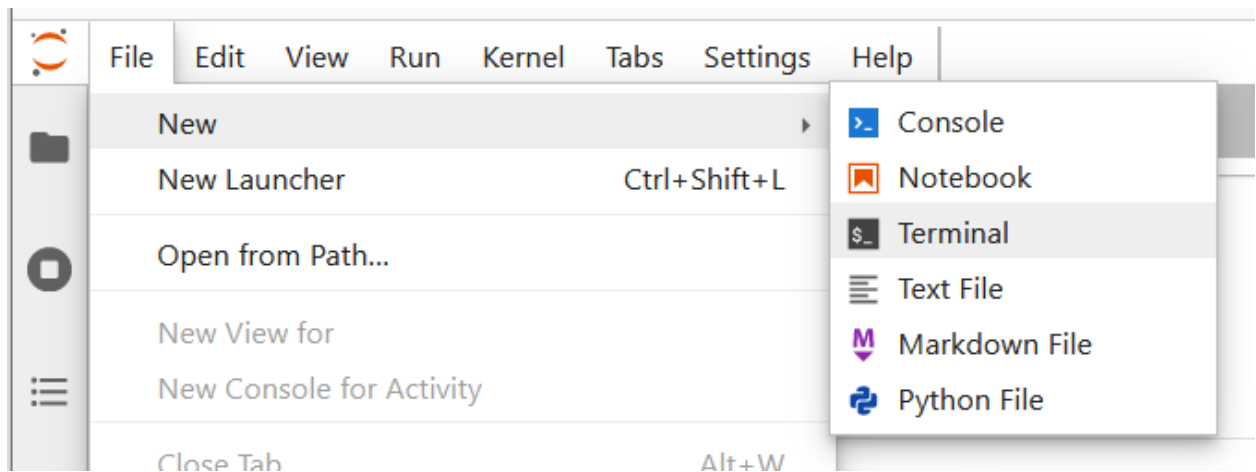
DevCloud account required (the files are there, and the FPGAs we will use).

Susannah Martin pulled together this lab and will walk us through it. These are my notes to help everyone follow along, or even try it out in advance so you can have your questions ready!

NOTE: the live demonstration of an FPGA usage will be done with the emulator, since the “compile” (place & route in ‘FPGA speak’) takes a little more than two hours (my most recent compile for this exercise took 2 hours 24 minutes). You are welcome to try the FPGA compile any time on your own – the instructions provide for that!

Use a browser –

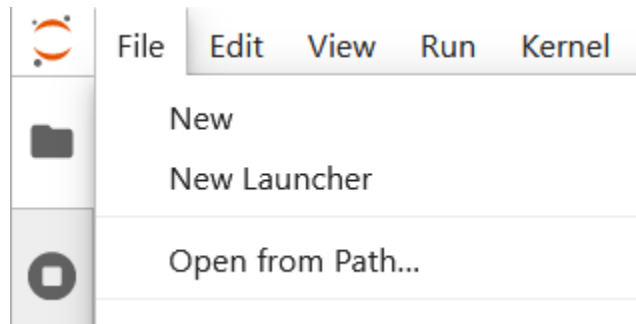
- (1) Get into a Jupyter notebook on DevCloud... visit <https://jupyter.oneapi.devcloud.intel.com/hub/login?next=/lab/tree/Welcome.ipynb?reset>
- (2) Click to sign in, click to start a server (if requested)
- (3) Open a terminal window (New->Terminal)



- (4) Type and run this command:

```
rsync -a /data/oneapi_workshop/xpublog/cppcon/FPGALab ~
```

- (5) Open the Jupyter notebook
`/FPGALab/fpga_dev_flow.ipynb`
using (File->Open from Path...)



Open Path

Path

`/FPGALab/fpga_dev_flow.ipynb`

Cancel

Open

(type in `/FPGALab/fpga_dev_flow.ipynb` and click Open)

NOTE: If it says 'file not found' – be sure you didn't have spaces before or after the file name when you pasted it into the dialog.



Save + Copy Paste Run Stop Refresh Next Previous Markdown

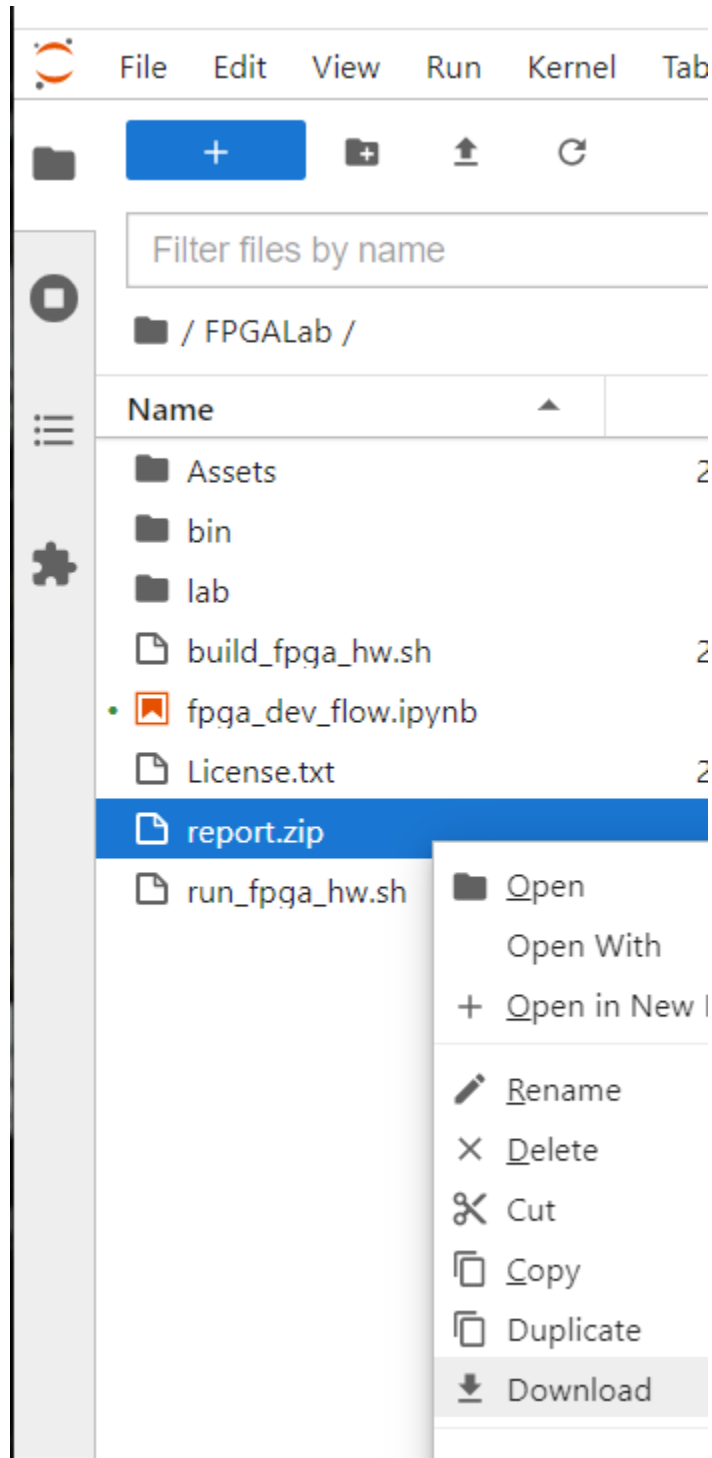
Lab: Practice the FPGA Development Flow

Sections

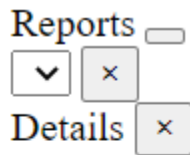
- [Development Flow for Using oneAPI with Intel® FPGAs](#)
- [Anatomy of a Compilation Command](#)
- [Stage 1: Emulation](#)
- [Stage 2: Optimization Report Generation](#)
- [Stage 3: Full Compile](#)
- [To Learn More](#)

(7) When you get to “**Now, let's examine that report file.**”

You can download the report.zip by right clicking and selecting Download.



- (8) Be sure to **EXTRACT** the ZIP file and view the extracted files. On Windows, if you simply browse into the ZIP file and try opening report.html you'll see something useless like this:



- (9) When compiling for the FPGA – you'll need to manually click on the qstat step, and click the Play button to refresh, and repeat here until it completes.

```
[16]: ! qstat -n -1
```

v-qsvr-1.aidevcloud:

Job ID	Username	Queue	Jobname	SessID	NDS	TSK	Req'd Memory	Req'd Time	S	Elap Time
--------	----------	-------	---------	--------	-----	-----	-----------------	---------------	---	--------------

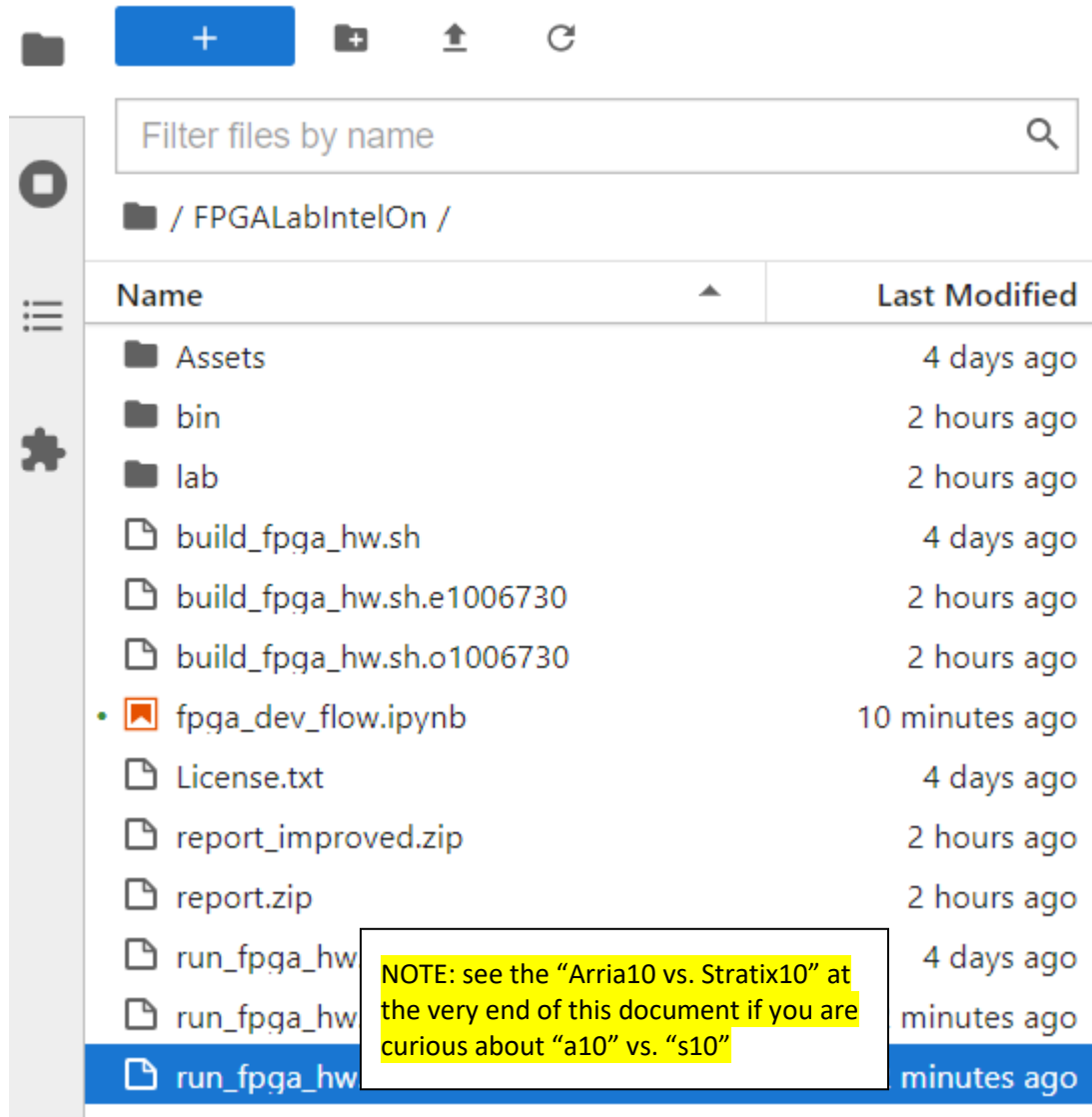
(10) To check on the compile – look in the build_fpga_hw_s10.sh.* files – o<number> for the stdout, and e<number> for stderr.

The screenshot shows the FPGA Lab IDE interface. The top menu bar includes File, Edit, View, Run, Kernel, Tabs, and Setting. Below the menu is a toolbar with icons for adding files, folders, and refreshing. The file explorer on the left shows the directory structure: / FPGALab /. The main pane displays a list of files and folders with columns for Name and Last Modified. The file 'build_fpga_hw_a10.sh.o10' is highlighted. A yellow callout box contains the following text:

NOTE: see the "Arria10 vs. Stratix10" at the very end of this document if you are curious about "a10" vs. "s10"

Name	Last Modified
Assets	10 hours
bin	seconds
bin-original	25 minutes
lab	4 minutes
build_fpga_hw_a10.sh	
build_fpga_hw_a10.sh.e10	
build_fpga_hw_a10.sh.o10	
build_fpga_hw_s10.sh	23 minutes

(11) When you proceed and run the final command (actually running on an FPGA), you'll need to open the output/error files to see the output (run_fpga_hw.s10.sh.* files)



Filter files by name

/ FPGALabIntelOn /

Name	Last Modified
Assets	4 days ago
bin	2 hours ago
lab	2 hours ago
build_fpga_hw.sh	4 days ago
build_fpga_hw.sh.e1006730	2 hours ago
build_fpga_hw.sh.o1006730	2 hours ago
• fpga_dev_flow.ipynb	10 minutes ago
License.txt	4 days ago
report_improved.zip	2 hours ago
report.zip	2 hours ago
run_fpga_hw	4 days ago
run_fpga_hw	minutes ago
run_fpga_hw	minutes ago

NOTE: see the "Arria10 vs. Stratix10" at the very end of this document if you are curious about "a10" vs. "s10"

(12) NOTE: The qstat command (which you can run over and over by clicking the cell, and then the run (play) button) will show an elapsed time of “--” while it is waiting for a node. If you get unlucky, and all the nodes are tied up – you’ll just have to wait. If you compiled for an Arria10 FPGA, then you’ll access to more nodes on DevCloud, than if you compiled for a Stratix10.

```
! qstat -n -1
```

v-qsvr-1.aidevcloud:

Job ID	Username	Queue	Jobname	SessID	NDS	TSK	Req'd Memory	Req'd Time	S	Elap Time	
1007872.v-qsvr-1.aidev	u64004	jupyterh	jupyterhub-singl	236112	1	1	94gb	04:02:00	R	00:35:33	s001-n024/1
1007882.v-qsvr-1.aidev	u64004	batch	run_fpga_hw.sh	--	1	2	--	06:00:00	Q	--	--

Yeah!

You have successfully (I hope) run code on an FPGA. Feel free to modify the code, and write your own, and explore FPGA programming via SYCL even more! Use the Jupyter notebook as a reference for the precise step-by-step commands to do this... esp. making note of the node types that are pre-installed with the compilation tools, and then the runtime libraries, to make your program work. All the software being used, is freely available via the oneAPI toolkits.

Stratix10 vs. Arria10 FPGAs

Susannah's notebook runs on Stratix10s, which are the nicest FPGAs on DevCloud. If you want to try Arria10 FPGAs (there are more of them to us), you need to change two things:

1. Change to this:
! qsub -l nodes=1:fpga_compile:ppn=2 -l walltime=24:00:00 -d . build_fpga_hw_**a10**.sh
from
! qsub -l nodes=1:fpga_compile:ppn=2 -l walltime=24:00:00 -d . build_fpga_hw_**s10**.sh
2. Change to this:
! qsub -l nodes=1:fpga_runtime:**arria10**:ppn=2 -d . run_fpga_hw_**a10**.sh
from
! qsub -l nodes=1:fpga_runtime:**stratix10**:ppn=2 -d . run_fpga_hw_**s10**.sh

Then go to the compile step, run it... wait, and then run the FPGA program. The Arria10s, in my experience, are more available. The Arria10 FPGAs are awesome, but the Stratix10s are more awesome (is that a thing?). 😊

NOTE: late breaking note... there is a dependency issue for compiling for Arria10. The compile will fail in under 3 minutes, and the output file will complain that PACSign is not found. We'll get this fixed... but until then, Stratix10 is the way to go although you may have to wait to see it run if people are camping out on the Stratix10 nodes (we've asked DevCloud to clear them if they can).

- james

James Reinders, Intel, engineer, james.r.reinders@intel.com