



# FCCM 2020 WORKSHOP: INTEL® FPGA CLOUDS FOR ACADEMIC RESEARCH AND TEACHING

Larry Landis – Intel Programmable Solutions Group

Eriko Nurvitadhi – Intel Programmable Solutions Group

Rui Ma – University of Texas, Austin

# AGENDA


**Part 1: Introduction to Intel academic program and FPGA clouds**

Part 2: Intel FPGA flows

Part 3: Deeper look at the clouds & demo

# INTEL FPGA ACADEMIC ECOSYSTEM

---

- 
- Engage research on Intel FPGAs
  - Train the next generation of FPGA Designers
  - Increase Intel FPGA presence in academia
  - Academic access to the latest generation of Intel FPGAs
  - Nurture the talent pipeline for Intel and our customers

# INTEL FPGA RESEARCH

## Academics

**Intel academic research programs**  
(e.g., ISRAs, single PIs)

**Research programs Intel involved in**  
(e.g., SRC JUMP)

**Internship program**

**Access to FPGA technologies**  
(e.g., HARP, DevCloud)

## Internal Research (PSG CTO, Intel Labs)

**DL workloads**

**Eval current FPGAs**

**Next-gen FPGAs**

**FPGA programmability**

**FPGAs in system**

...

## Broader Intel

**Engineering**  
(architecture, platform, tools, etc)

**Product planning**

**Marketing**

...

Transfers of innovations and technology

# INTEL FPGA ACADEMIC CLOUDS

Cloud access to Intel servers with FPGAs for academics

*FPGAs/SW tools already installed. Just login remotely. Ready to use!*

## HARP for long-term research

Hardware accelerator research program (HARP), originally offered cloud access to integrated (MCP) Xeon+FPGA

Now expanded to offer servers with FPGAs cards, hosted in Intel's Academic Compute Env.

Exclusively for long-term academic research (e.g., 1+ year PhD research)

## DevClouds for teaching and beyond

Offers servers with FPGA cards

Suitable for teaching (e.g., lab projects) and short-term research efforts. Move to HARP when research grows

And for short-term development projects in general (academic and industry)

# WHAT'S AVAILABLE TODAY

FPGA flows/framework	Devcloud	HARP
Traditional RTL flow	Y	Y
HLS Compiler	Y	Y
FPGA SDK for OpenCL	Y	Y
DPC++ (part of OneAPI)	Y	Upon Request
OpenVino (AI framework)	Y	Upon Request

## FPGA hardware

Intel Xeon with Arria 10 Programmable Acceleration Cards (PAC)

Intel Xeon with Stratix 10 Programmable Acceleration Cards (PAC)

Integrated Xeon+FPGA systems (HARP only)

# AGENDA

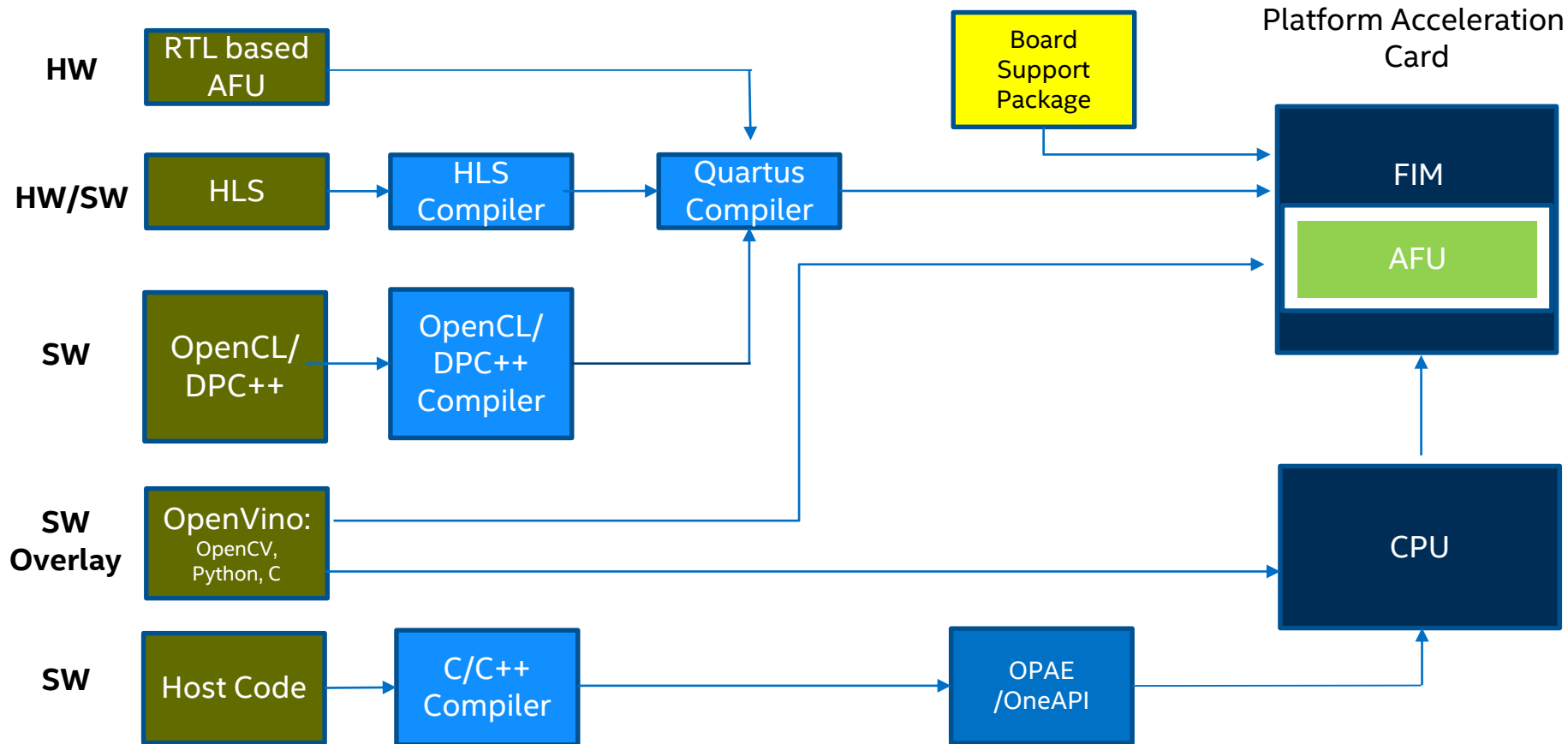
Part 1: Introduction to Intel academic program and FPGA clouds

**Part 2: Intel FPGA flows**

Part 3: Deeper look at the clouds & demo

# WORKING IN THE CLOUD: DEVELOPMENT FLOWS

Target  
Developer



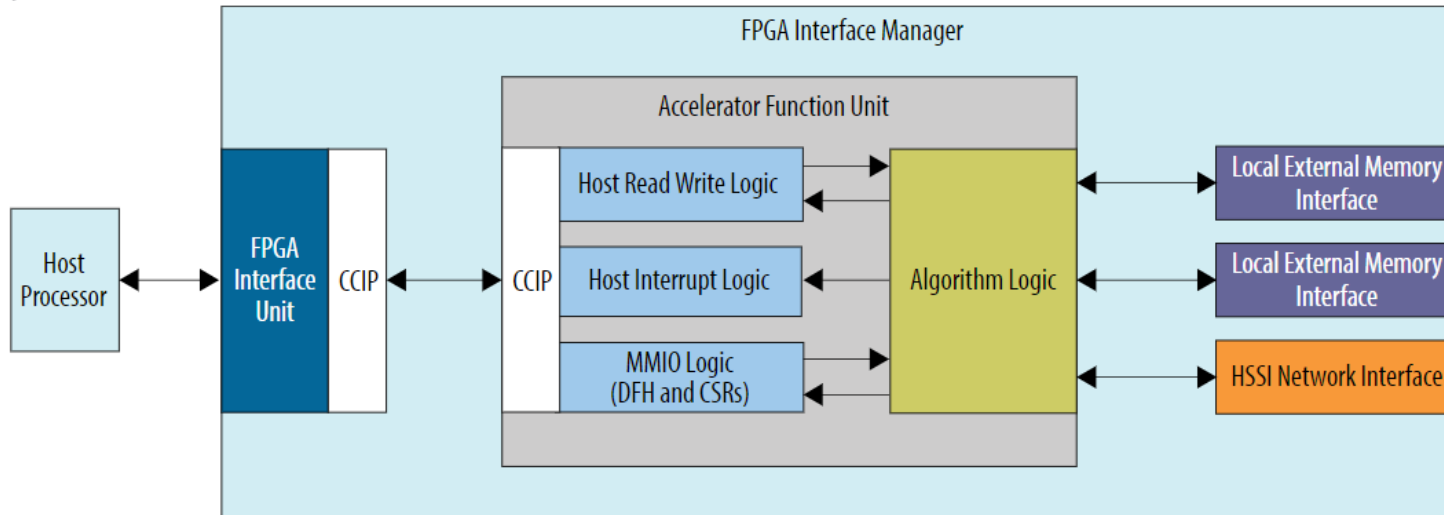


# RTL ACCELERATOR FUNCTIONAL UNIT (AFU)

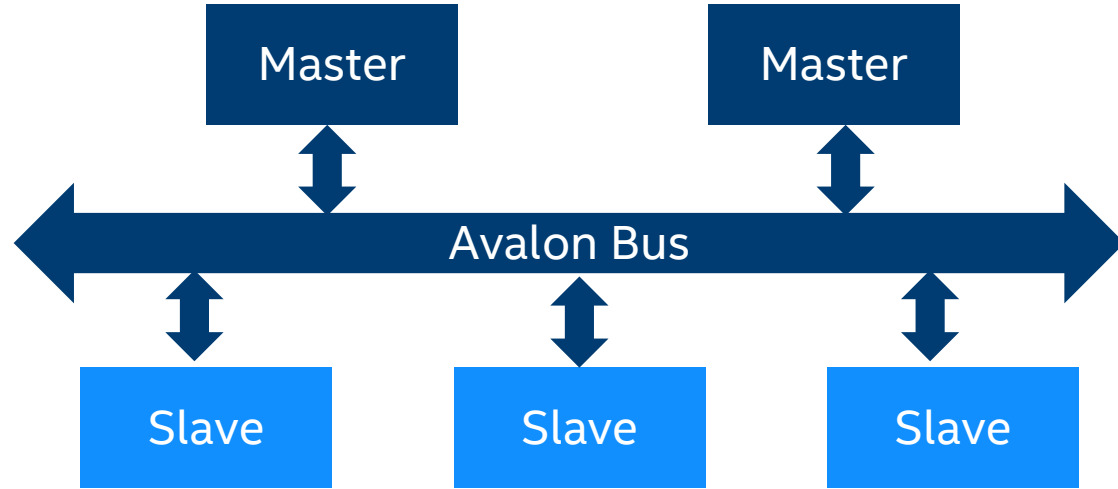
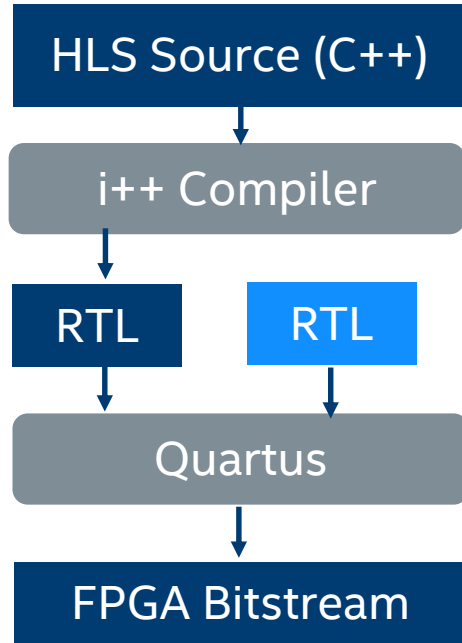
Intel Provides FPGA Interface Unit

Designer interfaces to Core Cache Interface Protocol (CCIP) spec

Algorithm Logic written in RTL



# HLS DESIGN FLOW



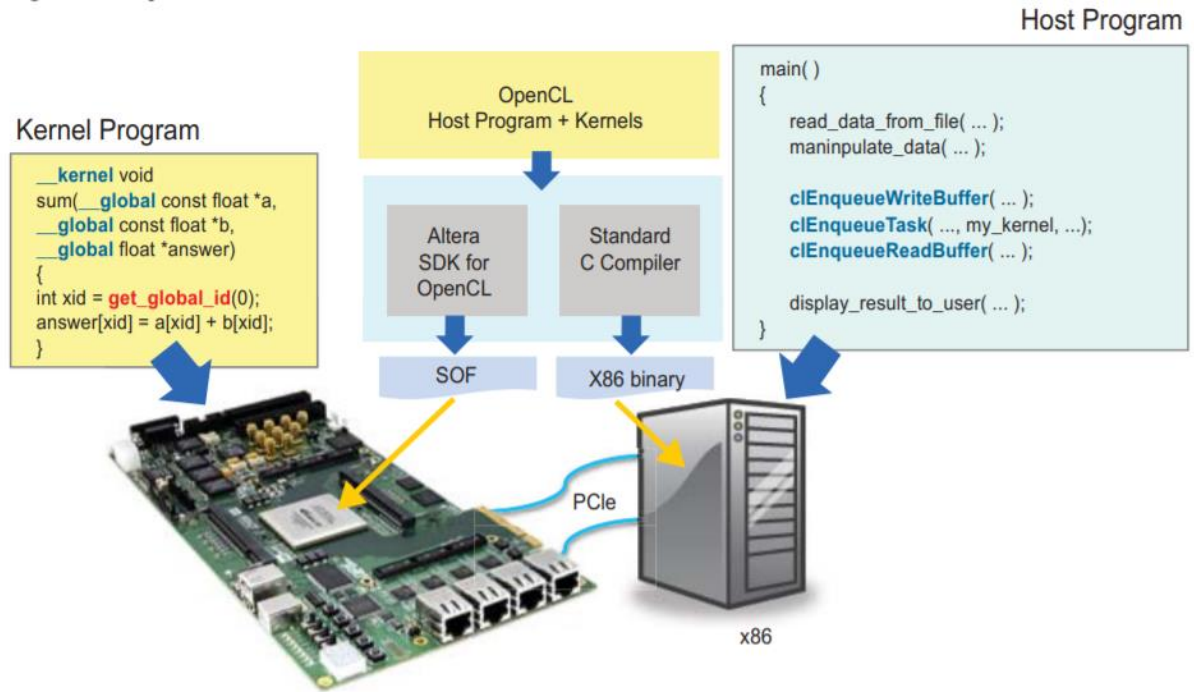
HLS is a module based flow – user designs IO interfaces / Board Support Package

# OPENCL FLOW

OpenCL Host written in C/C++ that runs on CPU

Kernels are C annotated with constructs to exploit FPGA parallelism and memory hierarchy

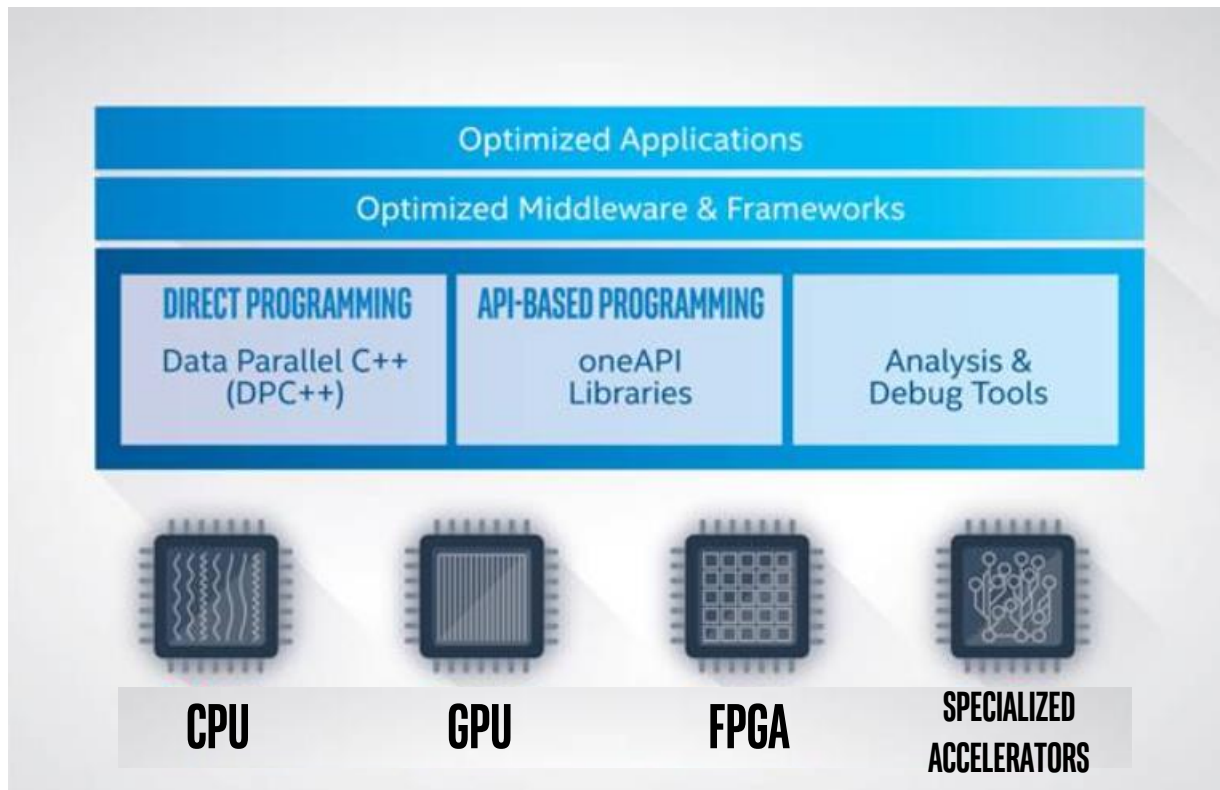
Compiler produces FPGA partial reconfiguration bitstream that runs with board support package



# ONEAPI/DPC++ FLOW

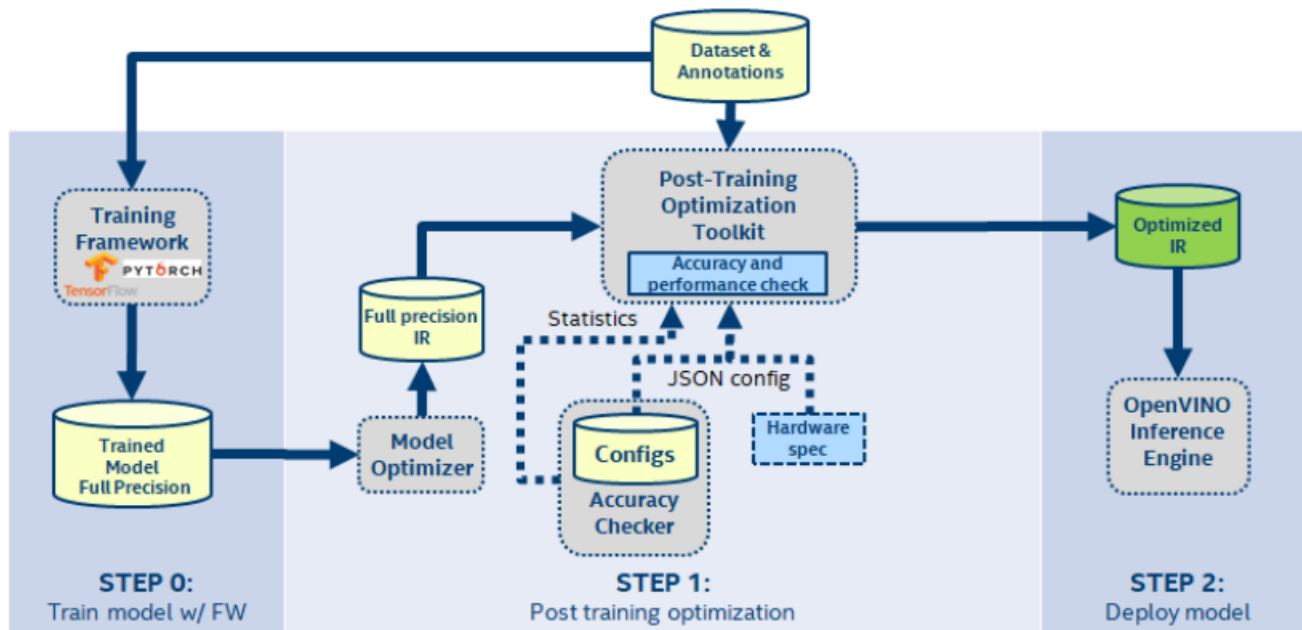
Tools and libraries  
to deploy  
applications across  
multiple  
architectures

DPC++ Open Source  
Language based on  
SYCL and  
community open  
source extensions



# OPENVINO – FPGA OVERLAY PREBUILT BITSTREAM

The Intel® Distribution of OpenVINO™ toolkit is a comprehensive toolkit for quickly developing applications and solutions that emulate human vision. Based on CNNs.



# AGENDA

Part 1: Introduction to Intel academic program and FPGA clouds

Part 2: Intel FPGA flows

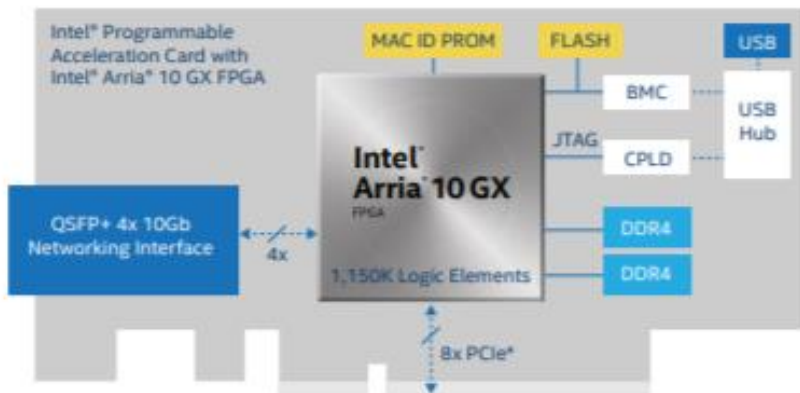
**Part 3: Deeper look at the clouds & demo**

# ARRIA 10 GX FPGA PROGRAMMABLE ACCELERATION CARD (PAC)



## Key Components and Interfaces

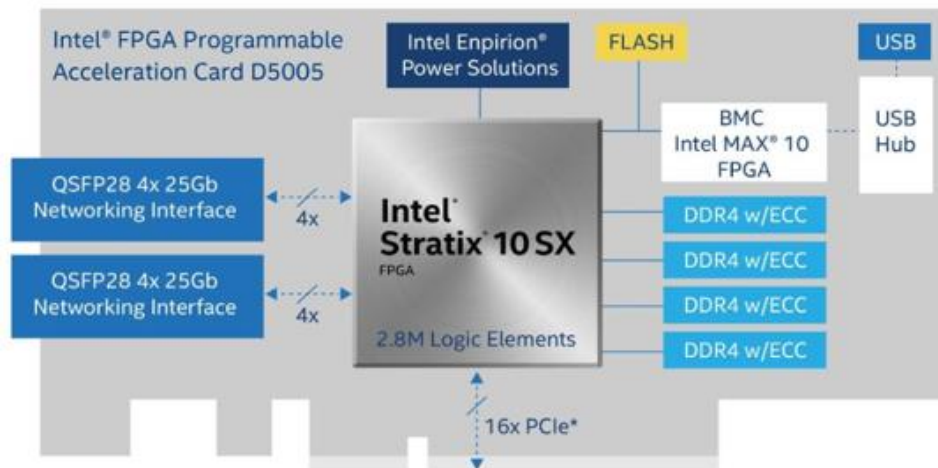
- Intel® Arria® 10 GX FPGA -1.15M LUTs
- 8 GB DDR4 memory banks (2 banks)
- 128 MB flash
- Quad small form factor pluggable (QSFP) interface speeds up to 40G
- PCI Express\* (PCIe\*) x8 Gen3 interface form factor
- Half-length card with standard (full height) and low profile (1/2 height) bracket options



# STRATIX 10 D5005 PAC

## Key Components and Interfaces

- Intel Stratix® 10 SX FPGA – 2.8M LUTs
- 32 GB DDR4 memory banks with error correction code (ECC) (4 banks, 2,400 Mbps)
- x2 quad small form factor pluggable (QSFP) with interface speeds up to 100G
- PCIe Gen3 x16 interface






# CLOUD RESOURCES

	Devcloud	HARP
CPU	225 x XEON Gold 6230	128 x Dual XEON Platinum 8280
Memory	192GB DDR4 2400MHz	768GB DDR4 2400Mhz
Arria 10 GX PAC	22	20
Stratix 10 D5005 PAC	2	30
Integrated Xeon+FPGA		23

# Website Access for the Intel DEVCLOUD



The screenshot shows the Intel DevCloud website interface. At the top, there's a blue header with the Intel Software Developer Zone logo on the left and a search bar, support link, sign-in option, and language selector on the right. The main content area has a dark, abstract background with glowing lines. It features the Intel DevCloud Workloads logo on the left, a large yellow text instruction to go to a specific URL, a central title for FPGA design development, a descriptive paragraph, a sign-up button, and a navigation menu at the bottom. A red circle and arrow highlight the 'FPGA' option in the navigation menu.

Go to URL <https://software.intel.com/en-us/devcloud/FPGA>

## FPGA DESIGN DEVELOPMENT AND WORKLOADS FOR HARDWARE ACCELERATION

Develop programmable solutions and validate your workloads on leading FPGA hardware with tools optimized for Intel® technology. Use this cloud solution in the classroom to support acceleration engineering curriculum.

[Sign Up for Beta](#)

Overview Data Center Edge **FPGA**

Intel Devcloud User Groups: Datacenter(/AI/OneAPI) Edge(IoT) **FPGA** ← Select the FPGA Cloud

# HARP: GETTING ACCESS + MORE DETAILS

## To get access

- Send email to: [IL\\_Academic\\_Res\\_Env@intel.com](mailto:IL_Academic_Res_Env@intel.com)
- Put email subject “[HARP] new account request”.
- Include a short (1 page max) research proposal
- Include the type of workloads you are planning to run

## More details

The following website offers information of available FPGA systems in HARP. It also provides detailed tutorials and examples on how to get started.

<https://wiki.intel-research.net/FPGA.html>

# DEMO

Rui Ma, PhD FPGA Research at University of Texas, Austin

Advisor: Professor Derek Chiou

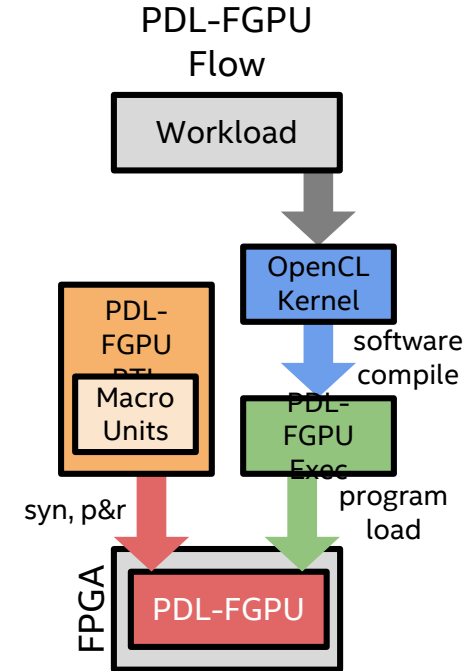
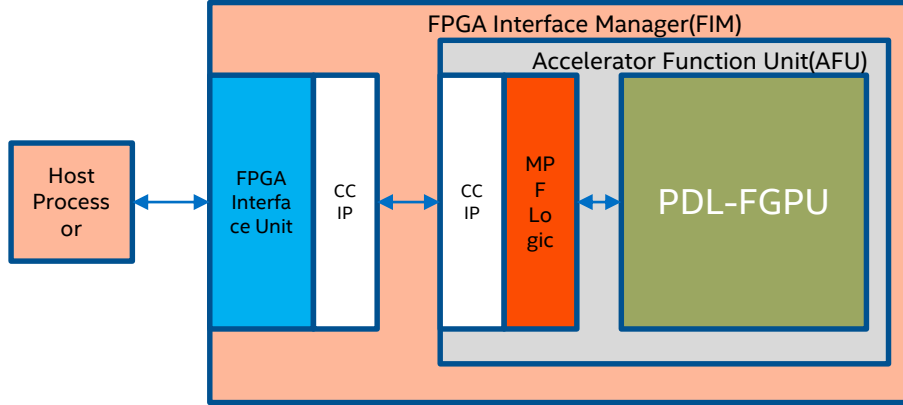
# MY PROJECT USING HARP: PDL-FGPU OVERLAY[1]

A SIMT architecture with domain specific specialization built on FPGA.

Runs binary compiled from OpenCL

- With a customized compiler based on LLVM
- Binary loaded to PDL-FGPU through CCI-P

Share memory with the host via CCI-P interface



[1] R. Ma et al., "Specializing FGPU for Persistent Deep Learning," 2019 29th International Conference on Field Programmable Logic and Applications (FPL), Barcelona, Spain, 2019, pp. 326-333.

# OVERALL EXPERIENCE USING HARP

Rich computation resources

Everything can be done remotely

- Do not need to physically set up the machines
- Best for working from home

OPAE and CCIP easy to use

- Portable
- Do not need to worry about PCIe

AFU is loaded to FPGA with partial reconfiguration technique

- Be careful about the register initialization

# STEPS TO LOGIN TO INTEL HARP

ssh to HARP cluster

- `$ ssh -X <username>@ssh-iam.intel-research.net`

Configure the environment

- `$ source /export/fpga/bin/setup-fpga-env fpga-pac-s10`

Request an FPGA node

- `qsub-fpga`

HARP docs: <https://wiki.intel-research.net/FPGA.html>

# STEPS TO BUILD FPGA AFU

## Build FPGA program

- Generate build directory
  - `$ afu_synth_setup -s ./RTL/sources.txt build_synth`
- Launch a batch job to compile the RTL source
  - `$ cd build_synth`
  - `$ qsub-synth`

## Build host program

- `$ cd sw`
- `$ make`

OPAE User Guide: <https://www.intel.com/content/www/us/en/programmable/documentation/bfr1522087299048.html>

OPAE Tutorial: <https://github.com/OPAE/intel-fpga-bbb/tree/master/samples/tutorial>



# PROGRAM FPGA AND RUN THE HOST PROGRAM

## Program the FPGA

- \$ cd build\_synth
- \$ fpgaconf cci\_mpf\_hello.gbs

## Example run: RNN

- 256 iterations
- Floating point
- Hidden layer size (1024)

```
[marui9633@vsl089 sw]$ ./cci_mpf_hello cram.mif krnl_ram.mif
page size:4096
write base address:2aaaaac00000
clean cache: 0
load kernel
load kernel params
read kernel
start kernel
total time:0.00423205s
```

