FPGAs are very efficient... but hard to program?

Programmability (Ease of use) vs **Efficiency**

CPU, GPU, FPGA, ASIC

Can FPGA learn from CPU style approach to offer improved software programmability?
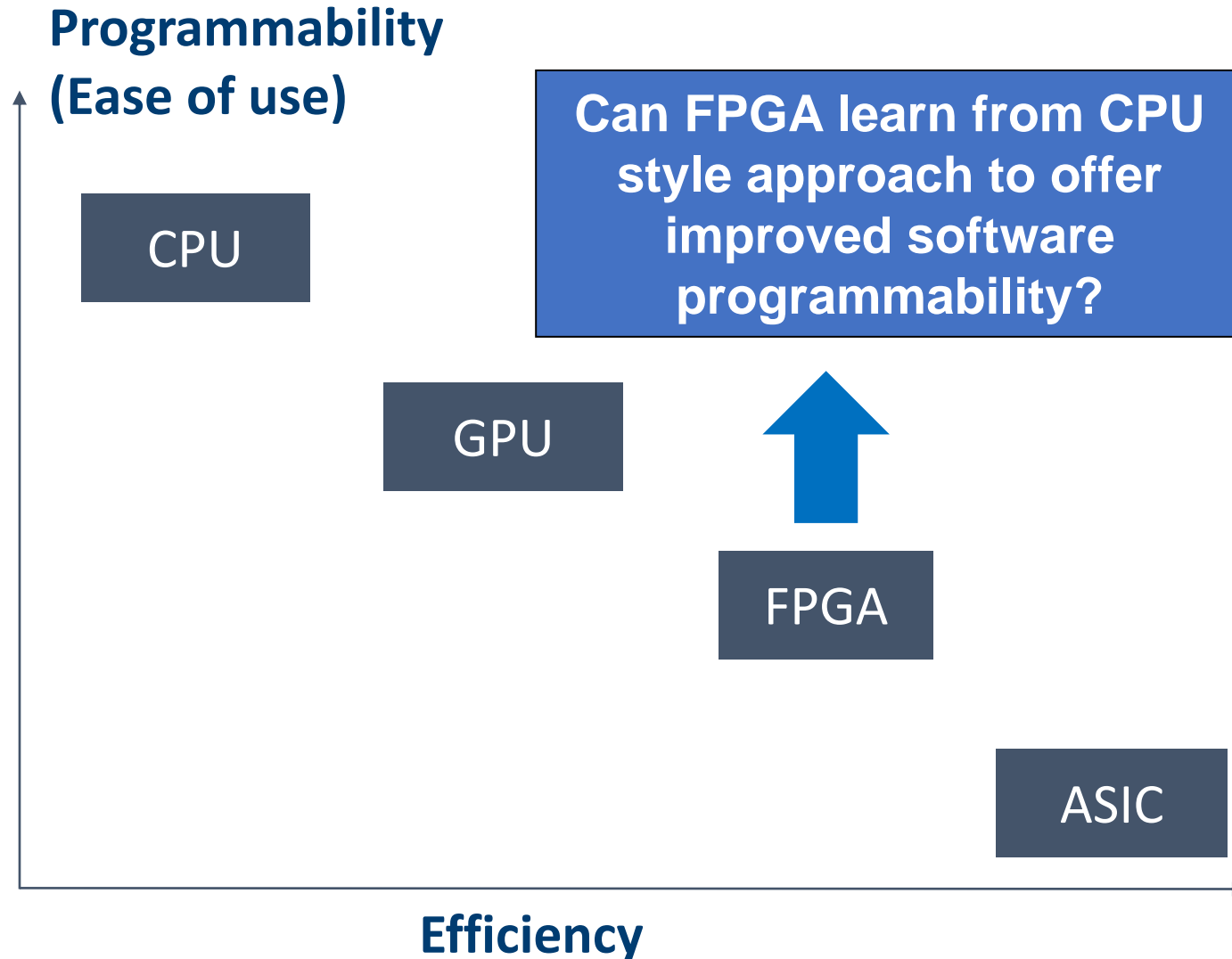
My Project using HARP: PDL-FGPU Overlay[1]

- 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

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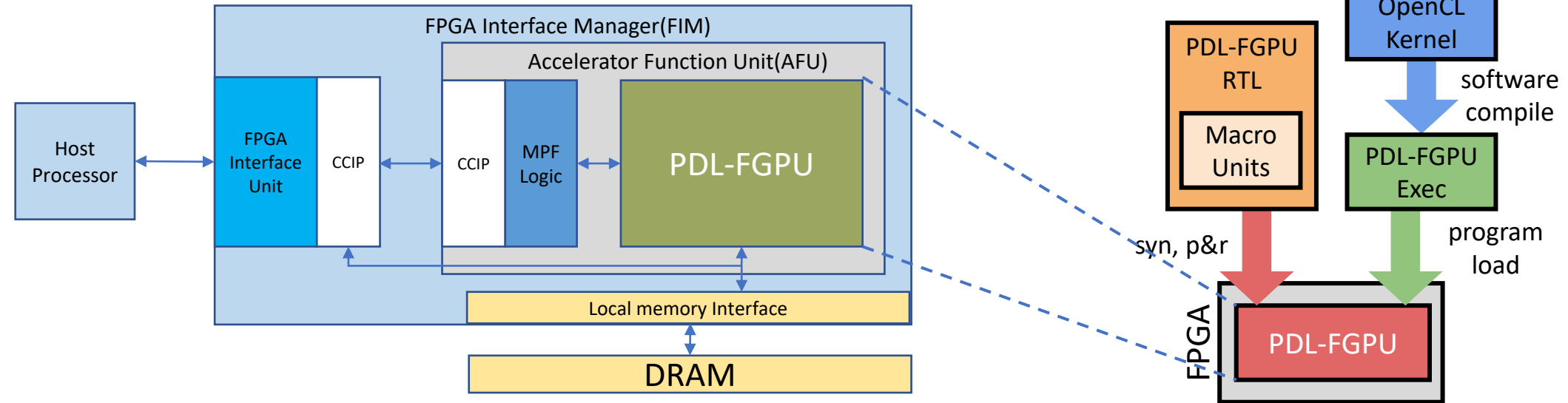
[1] R. Ma *et al.*, "Specializing FGPU for Persistent Deep Learning," *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*, Barcelona, Spain, 2019, pp. 326-333.

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 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
```