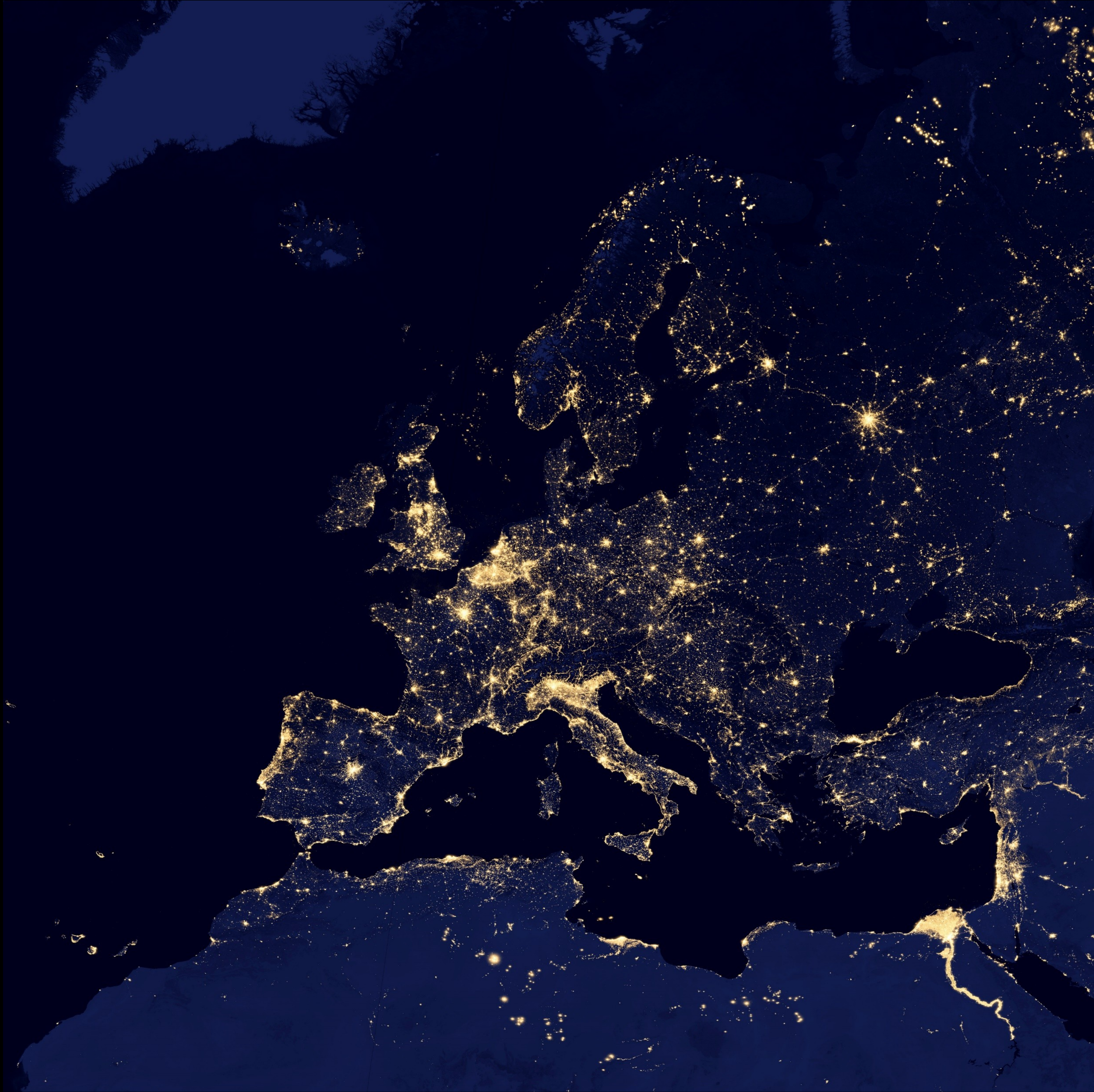
Distributed Operation Set Architectures for low-latency ML inference using FPGAs

Introducing DOSA

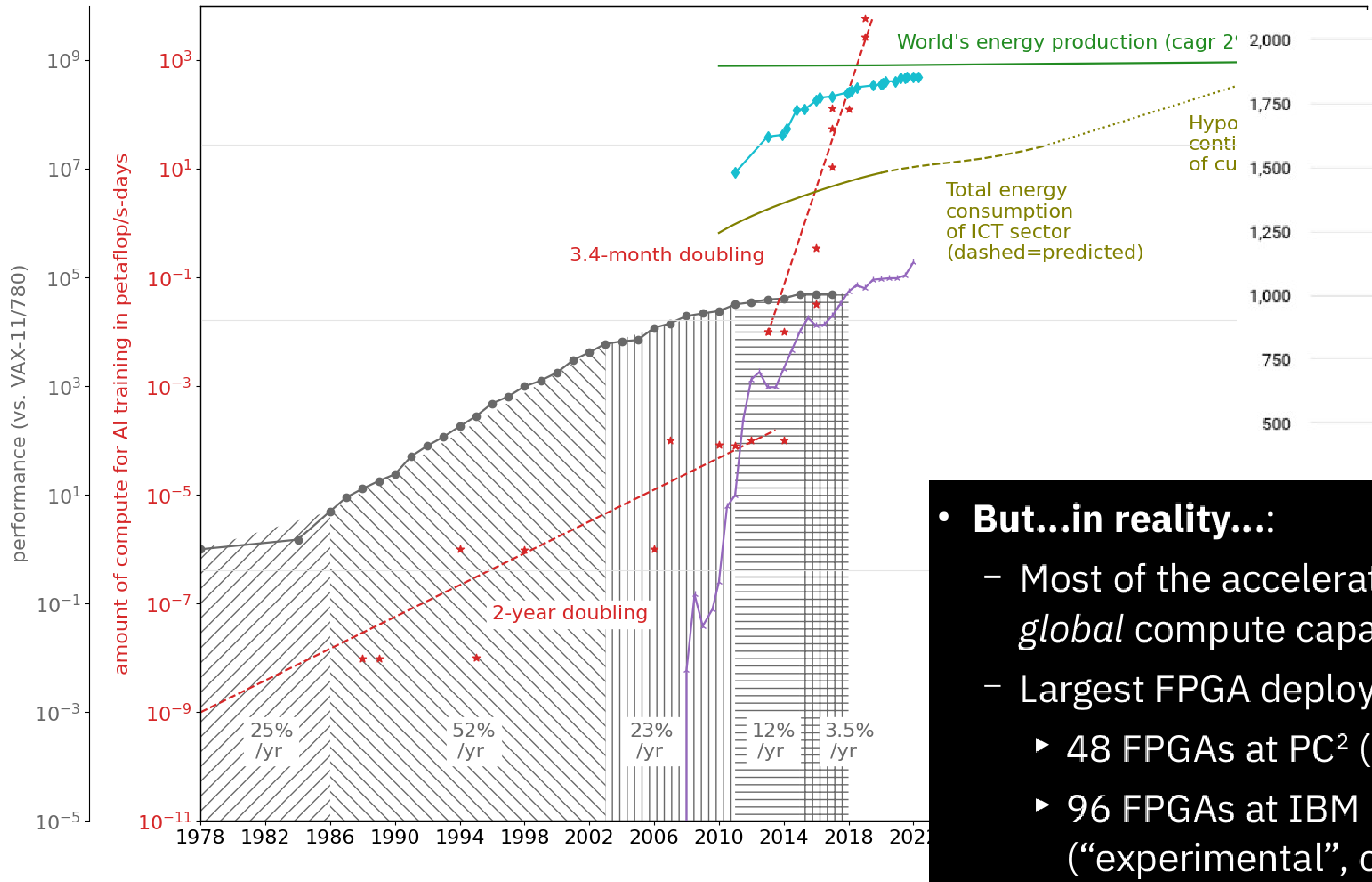
—

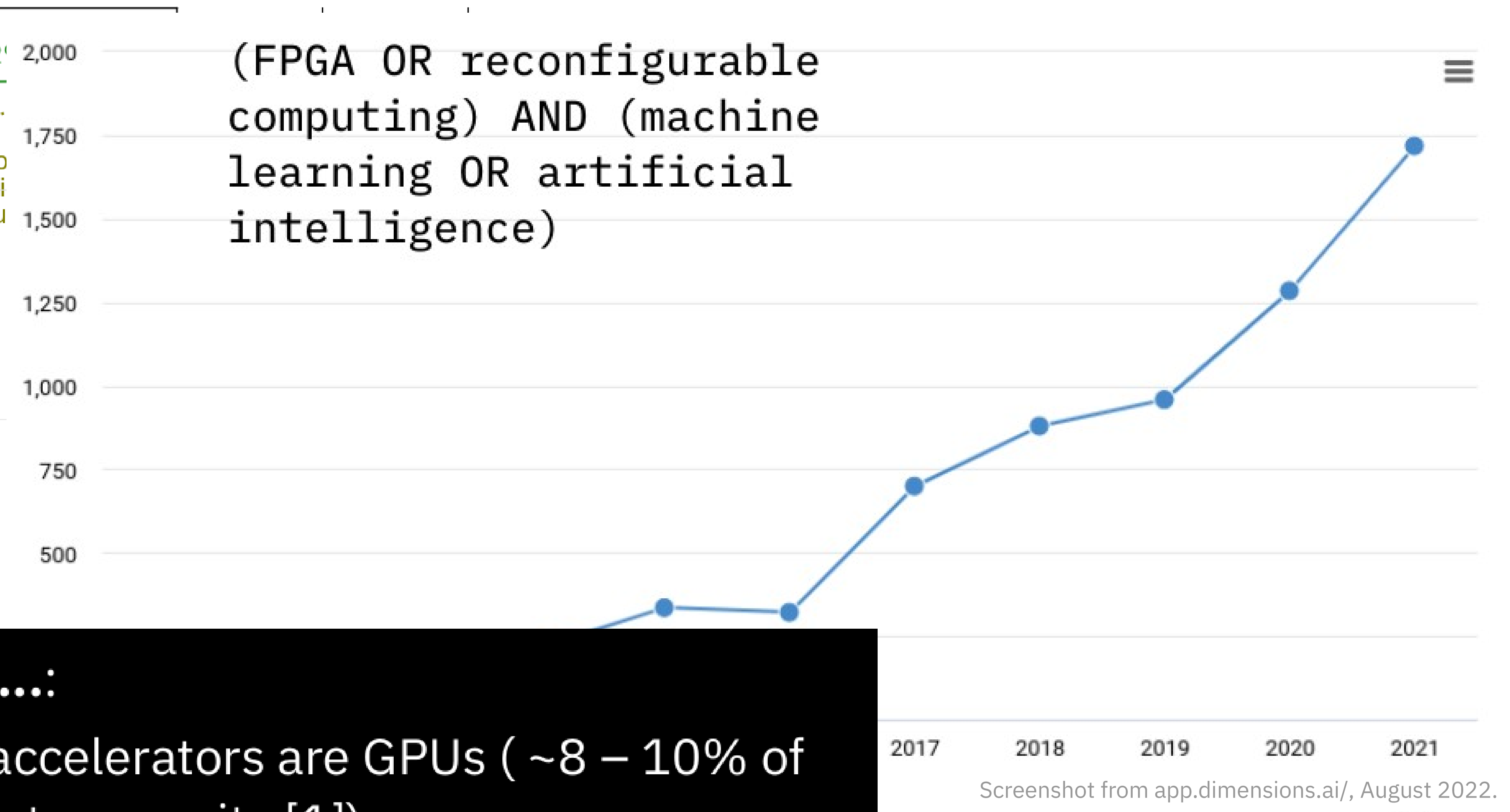
Dr. Burkhard Ringlein
IBM Research – Europe
Zurich, Switzerland

EVERST + DAPHNE: Workshop
HiPEAC 2024
2024-01-19



I guess, we all know why we are here...



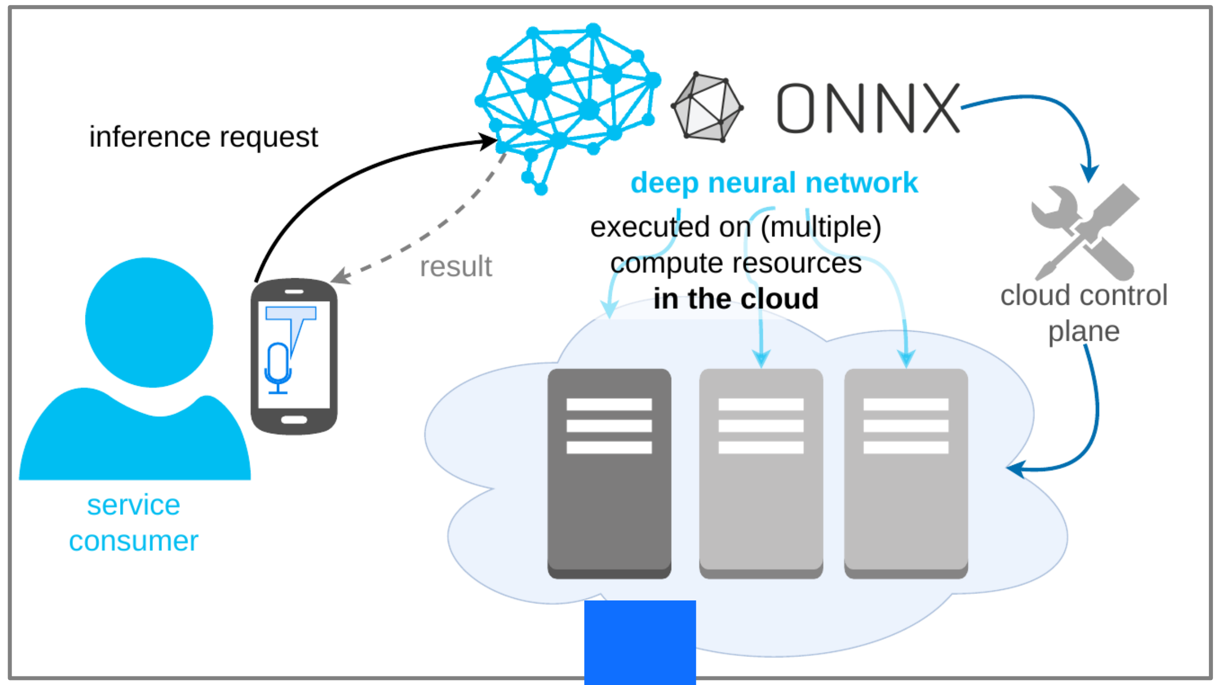


Screenshot from app.dimensions.ai/, August 2022.

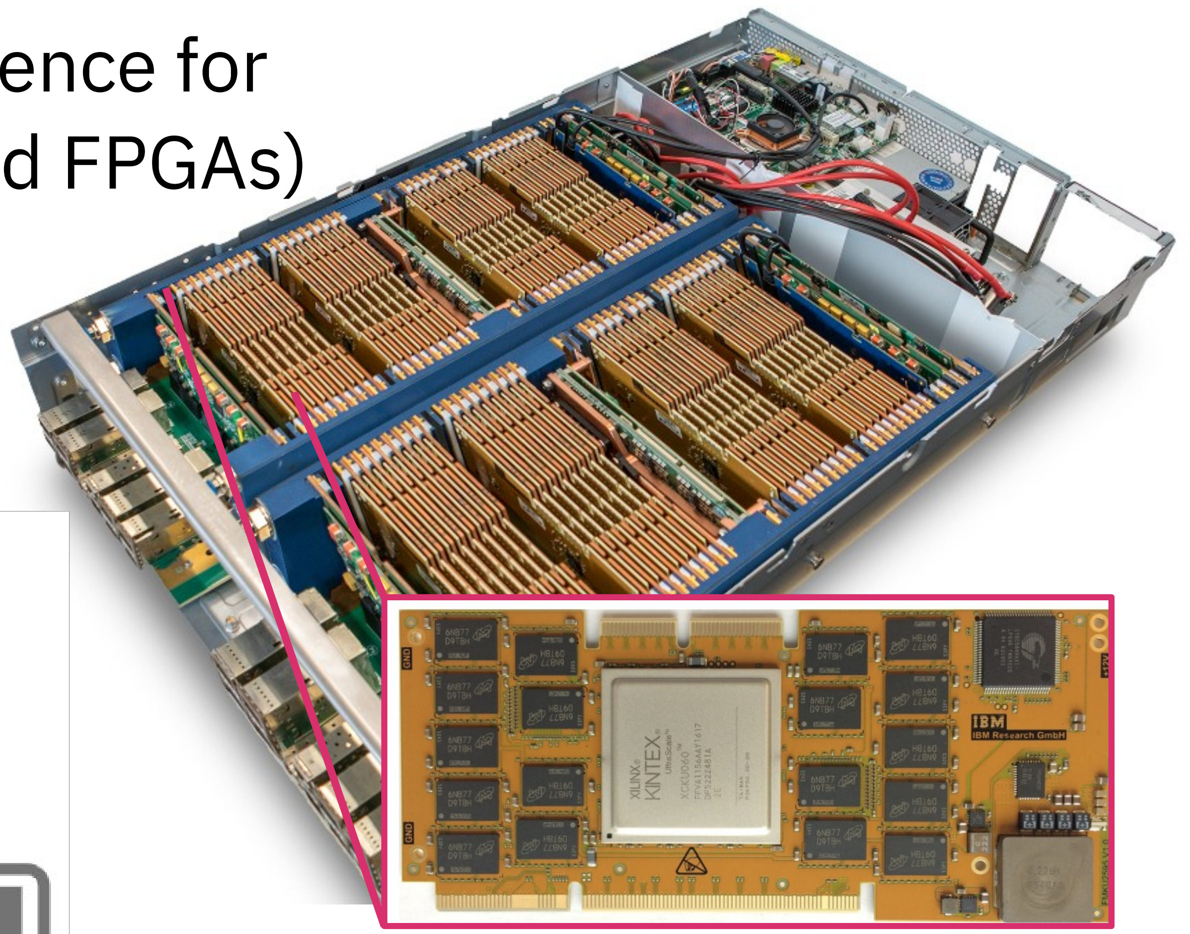
- **But...in reality...:**
  - Most of the accelerators are GPUs (~8 – 10% of *global* compute capacity [1])
  - Largest FPGA deployments (AFAIK):
    - 48 FPGAs at PC² (“production”, Alveos)
    - 96 FPGAs at IBM Research Zurich (“experimental”, cloudFPGA platform)
    - Cloud services hard to measure, but no large growth observable...
- So, why aren’t there more FPGAs used for ML & HPC?

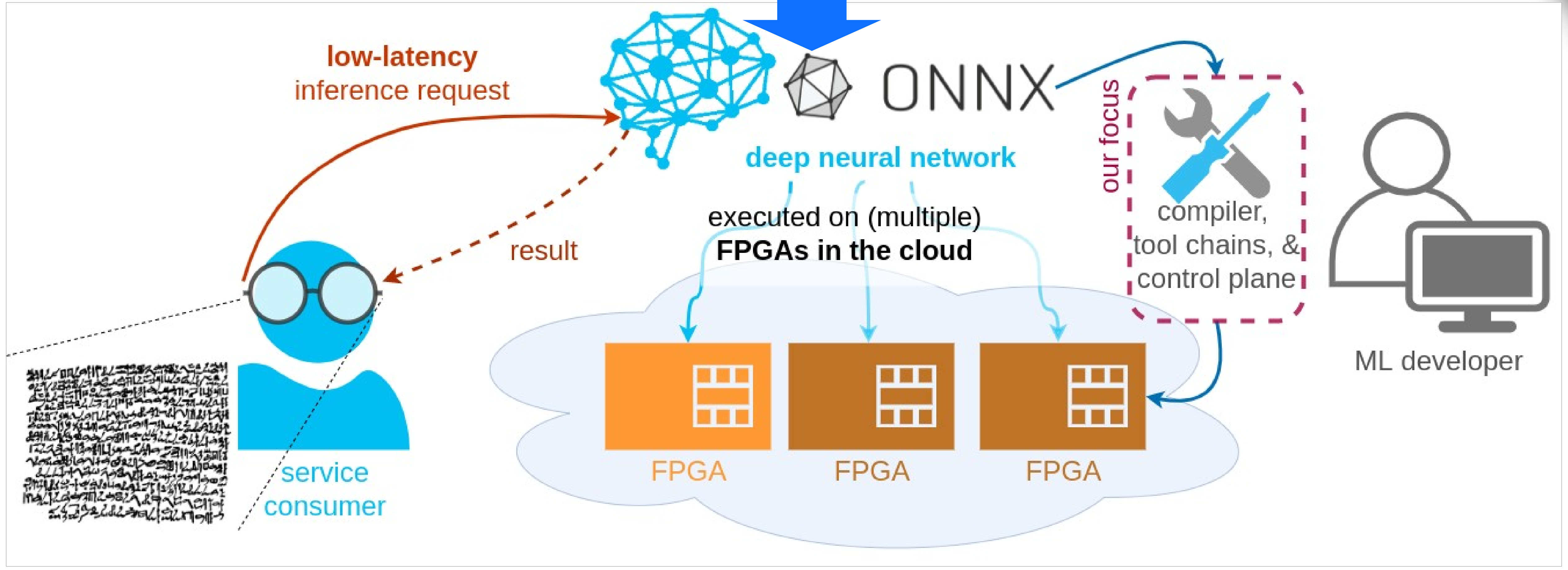
Agenda: Is one-click DNN to distributed FPGA compilation possible?

Accelerated Inference-as-a-Service is presented here as abstracted use case type (also for the use cases within EVEREST)



Target platform:
IBM cloudFPGA
(as reference for distributed FPGAs)

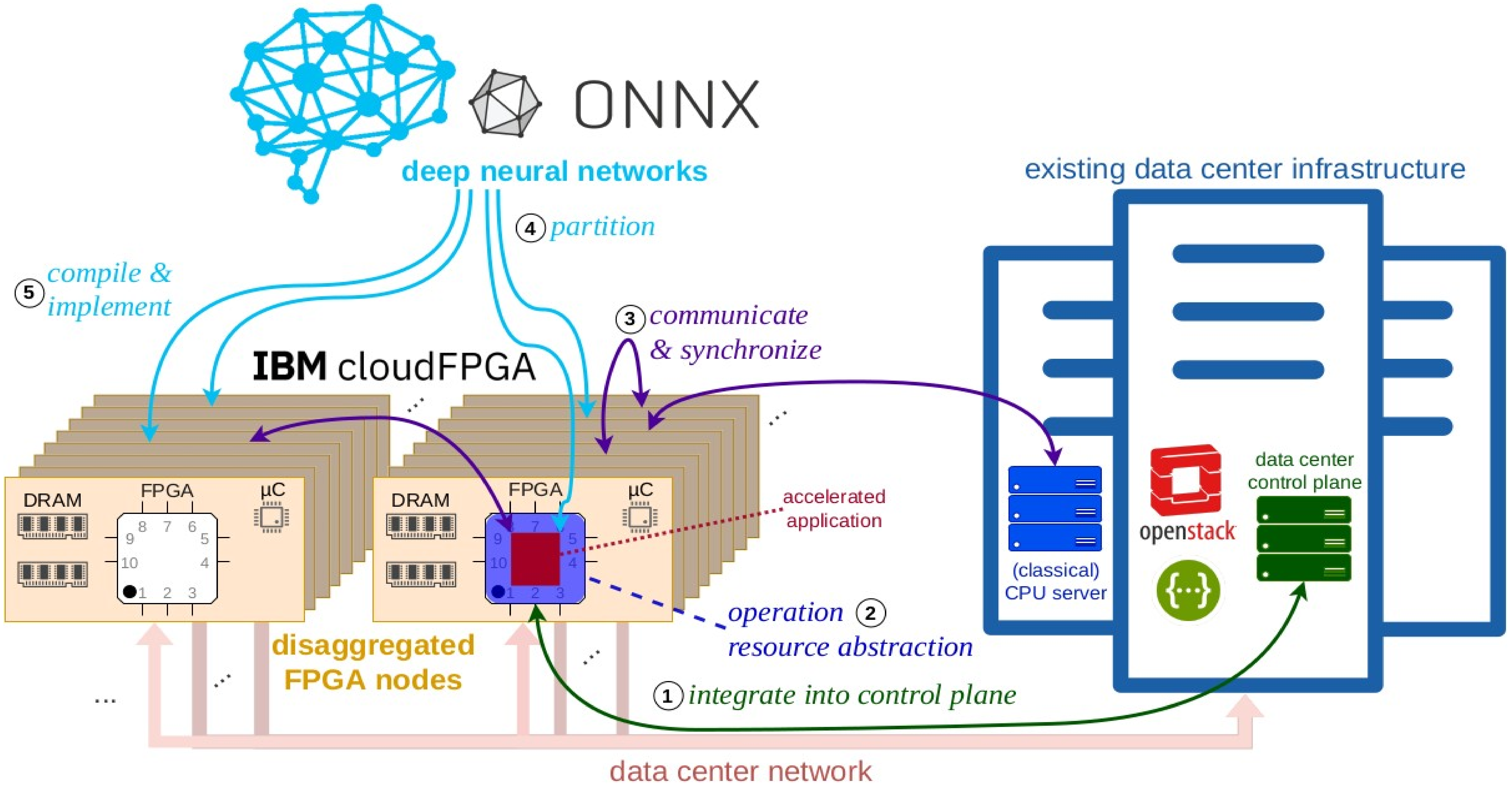




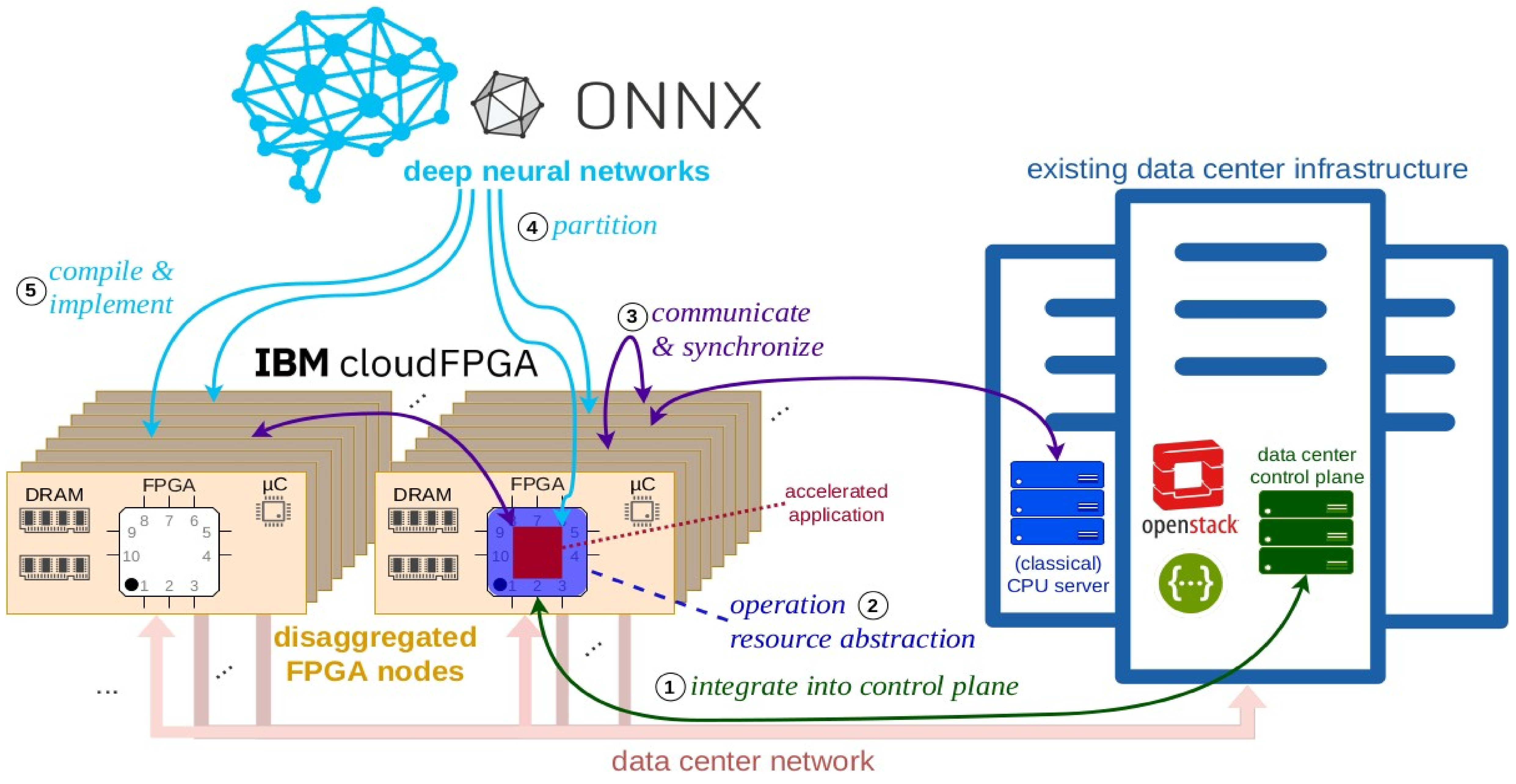
low-latency inference request
result
ONNX
deep neural network
executed on (multiple) **FPGAs in the cloud**
our focus
compiler, tool chains, & control plane
ML developer
FPGA
FPGA
FPGA
service consumer

→ **In this presentation**, I close the gaps between ML representations (ONNX) and distributed FPGAs (cloudFPGA).

Overview: 5 necessary steps to map ONNX to cloudFPGA



Overview: 5 necessary steps to map ONNX to cloudFPGA



4 + 5

→ Our focus today! DOSA

Published in: **IEEE CAL 2023, EDGE 2023**

Open-source release: https://github.com/cloudFPGA/DOSA

Developed within H2020 EVEREST

3 Published in: **FCCM 2020, H2RC 2020**

Open-source release: https://github.com/cloudFPGA/ZRLMPI

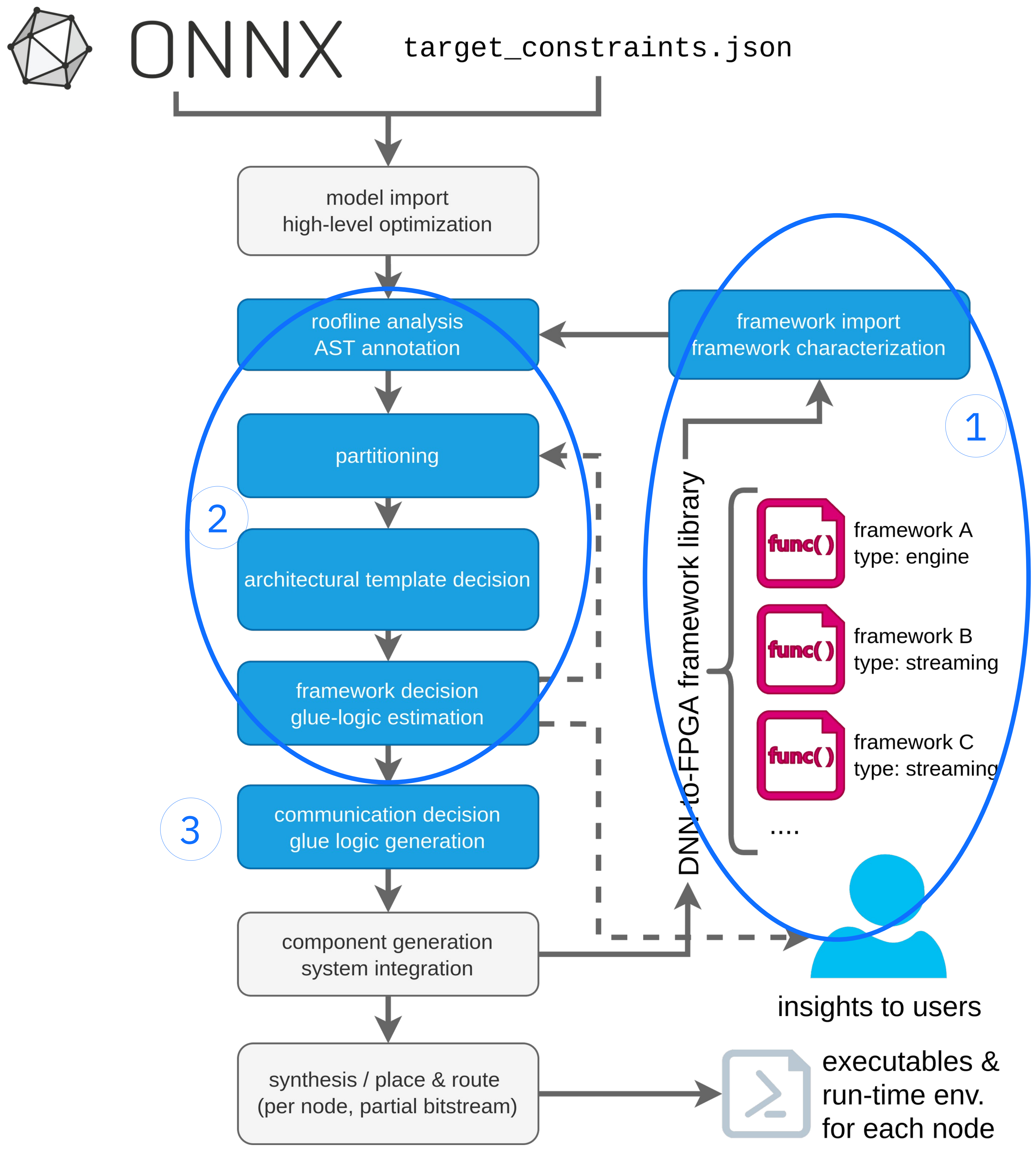
1 + 2 Published in: **FPL 2019, CLOUD 2021**

Open-source release: https://github.com/cloudFPGA/cFDK

DOSA, automated compilation of DNN to distributed FPGAS

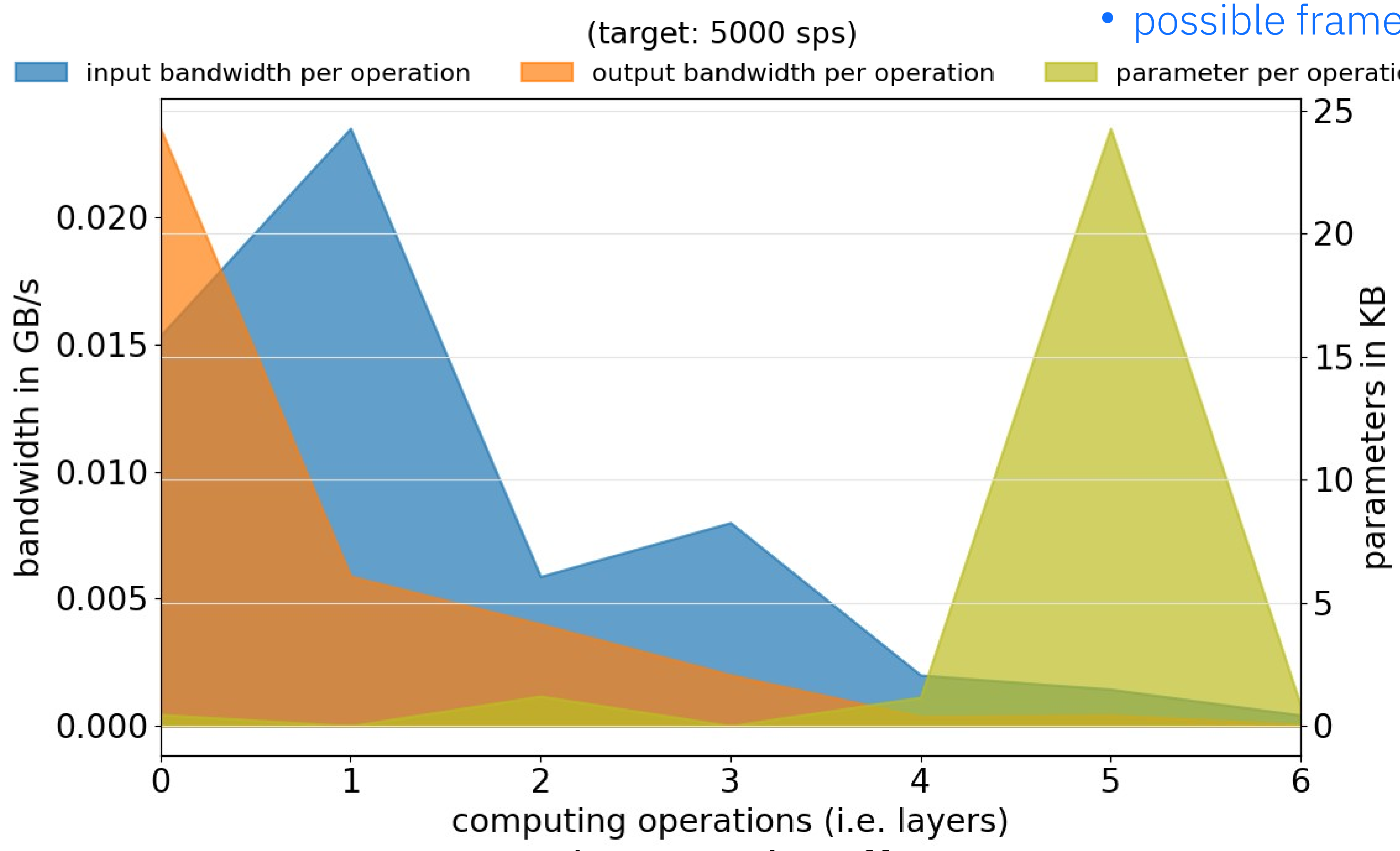
- One tool to cover large solution space, different optimizations, and major standards
  - Avoids “re-inventing the wheel”: composes open source tools: e.g. TVM, hls4ml, haddoc2, VTA
  - But: combines them in an optimal way
  - Based on roofline analysis and framework characterization
  - Decision based on performance constraints

  ➔ Automated re-use with *organic-compiler concept* and *operator set generators*
- Automatic partitioning: Model & data parallelism



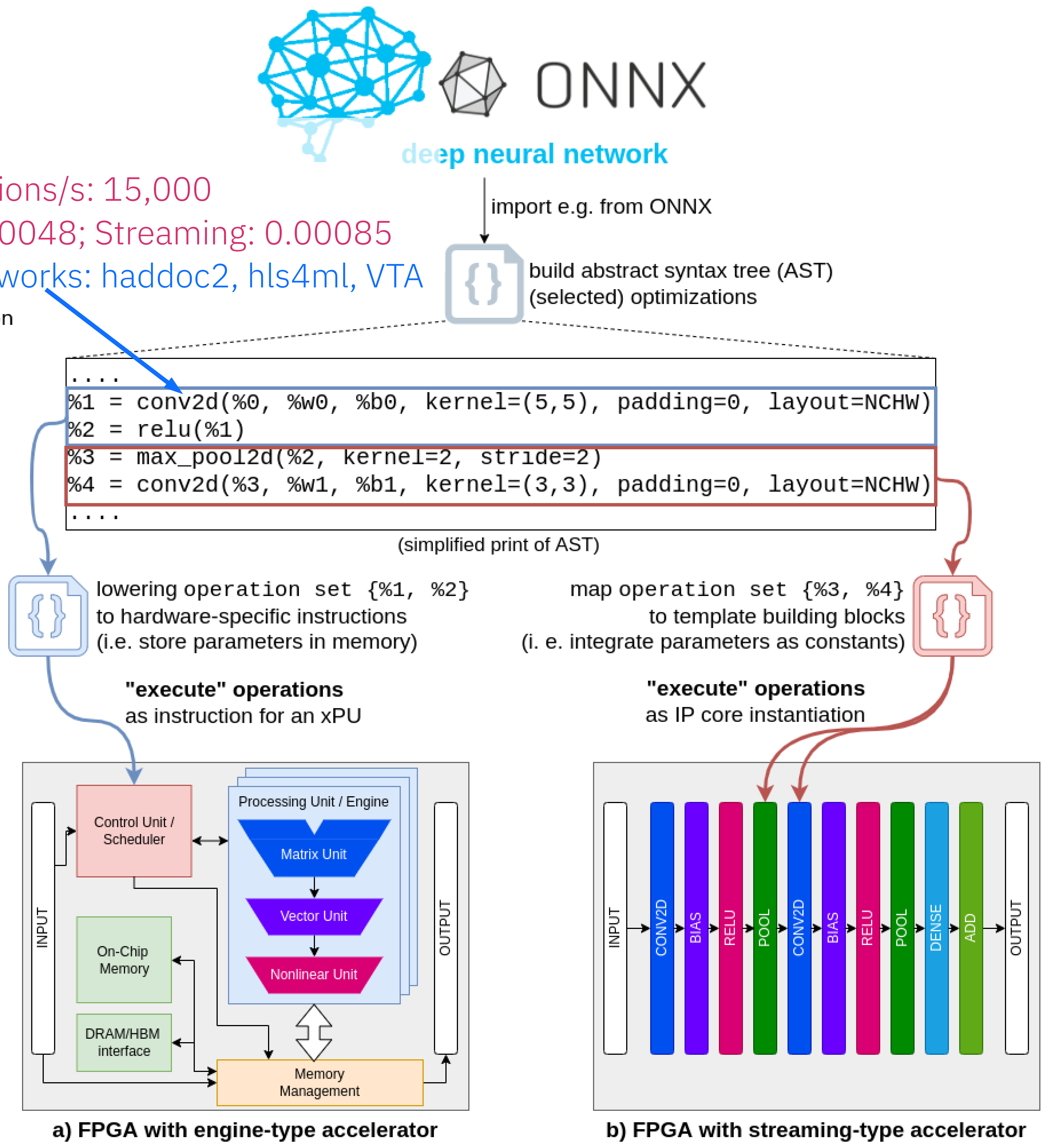
Combining Streaming- and Engine-type designs

conv2d:
- Required iterations/s: 15,000
- OI: Engine: 0.00048; Streaming: 0.00085
- possible frameworks: haddoc2, hls4ml, VTA



→ *best trade-off?*

Enabler: Implementation abstraction using the Operation Set Architecture



```
....
%1 = conv2d(%0, %w0, %b0, kernel=(5,5), padding=0, layout=NCHW)
%2 = relu(%1)
%3 = max_pool2d(%2, kernel=2, stride=2)
%4 = conv2d(%3, %w1, %b1, kernel=(3,3), padding=0, layout=NCHW)
....
```

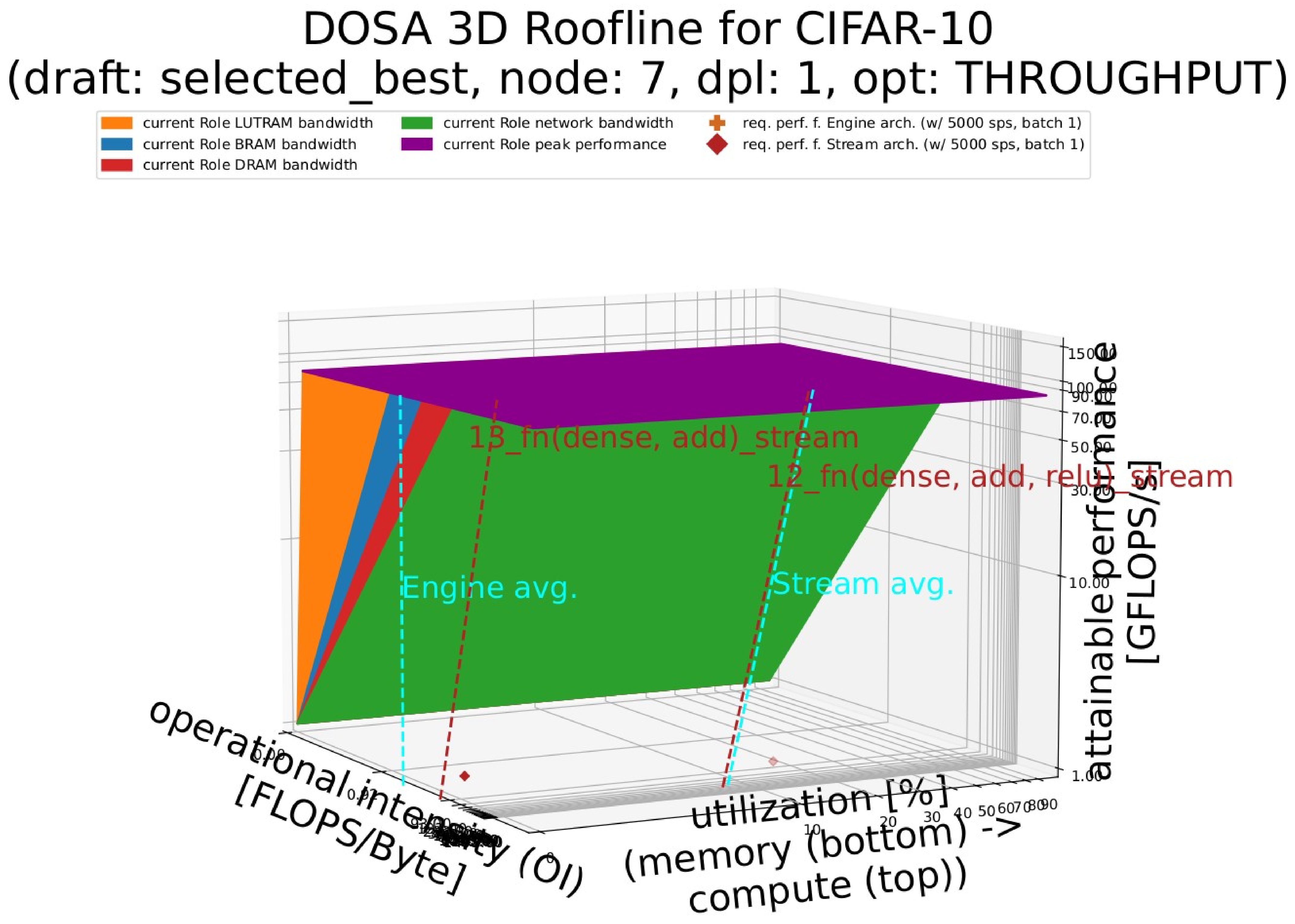
a) FPGA with engine-type accelerator

b) FPGA with streaming-type accelerator

Example CNN from https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

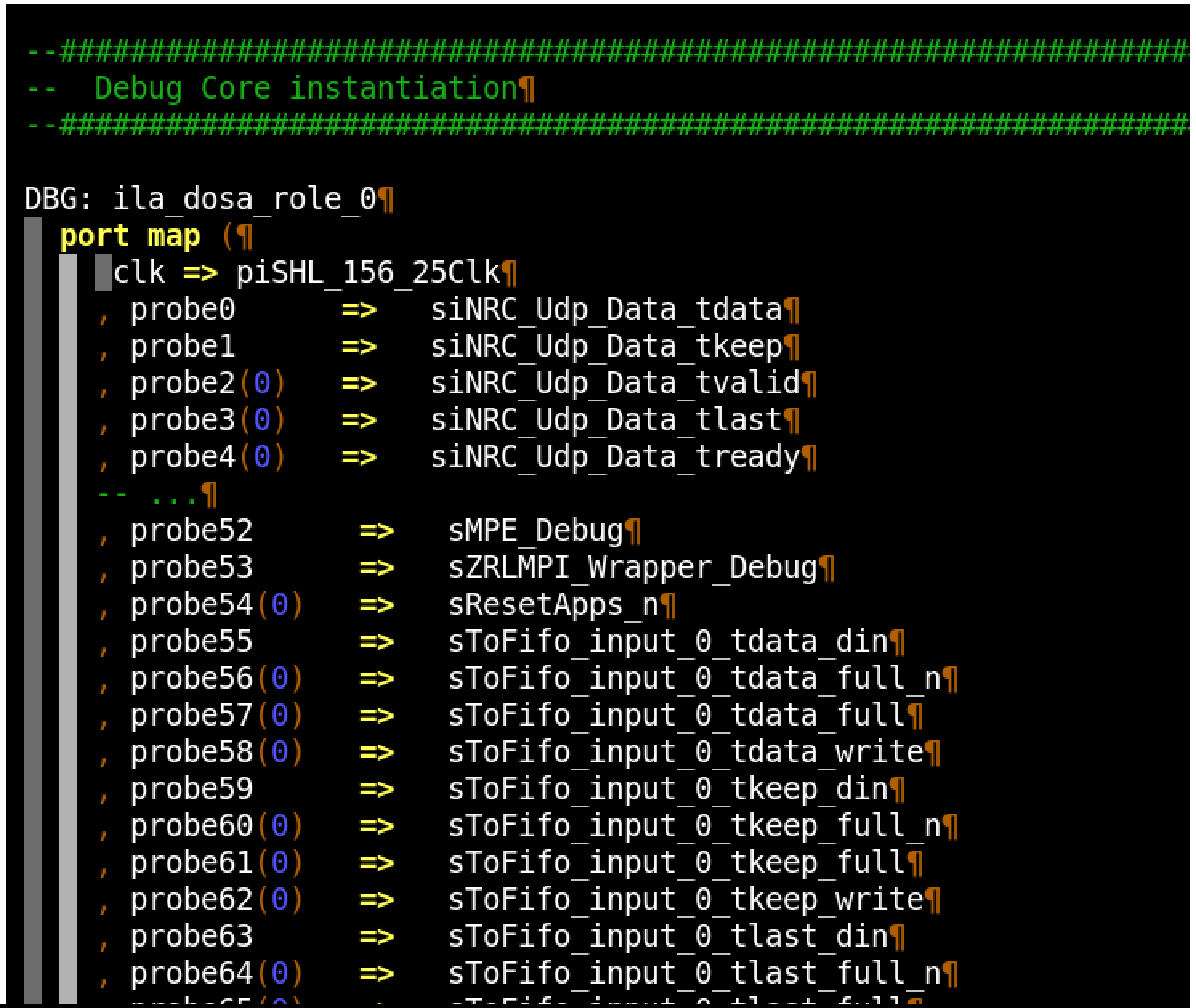
DSE based on “3D Rooflines”

- Multi-dimensional space:
  - Resources (i.e. costs)
    - but details matter: LUT, LUTRAM, BRAM, DSP, FF
  - Latency vs throughput
  - Costs of combining different libraries
- “hardware-aware”/”system-aware” optimization:
  DOSA doesn’t consider solutions that would violate the Roofline model
- Multiple runs with different hyper-parametern (e.g. “osg_look_ahead”)
- Solution with lowest costs that fulfills target criteria is chosen

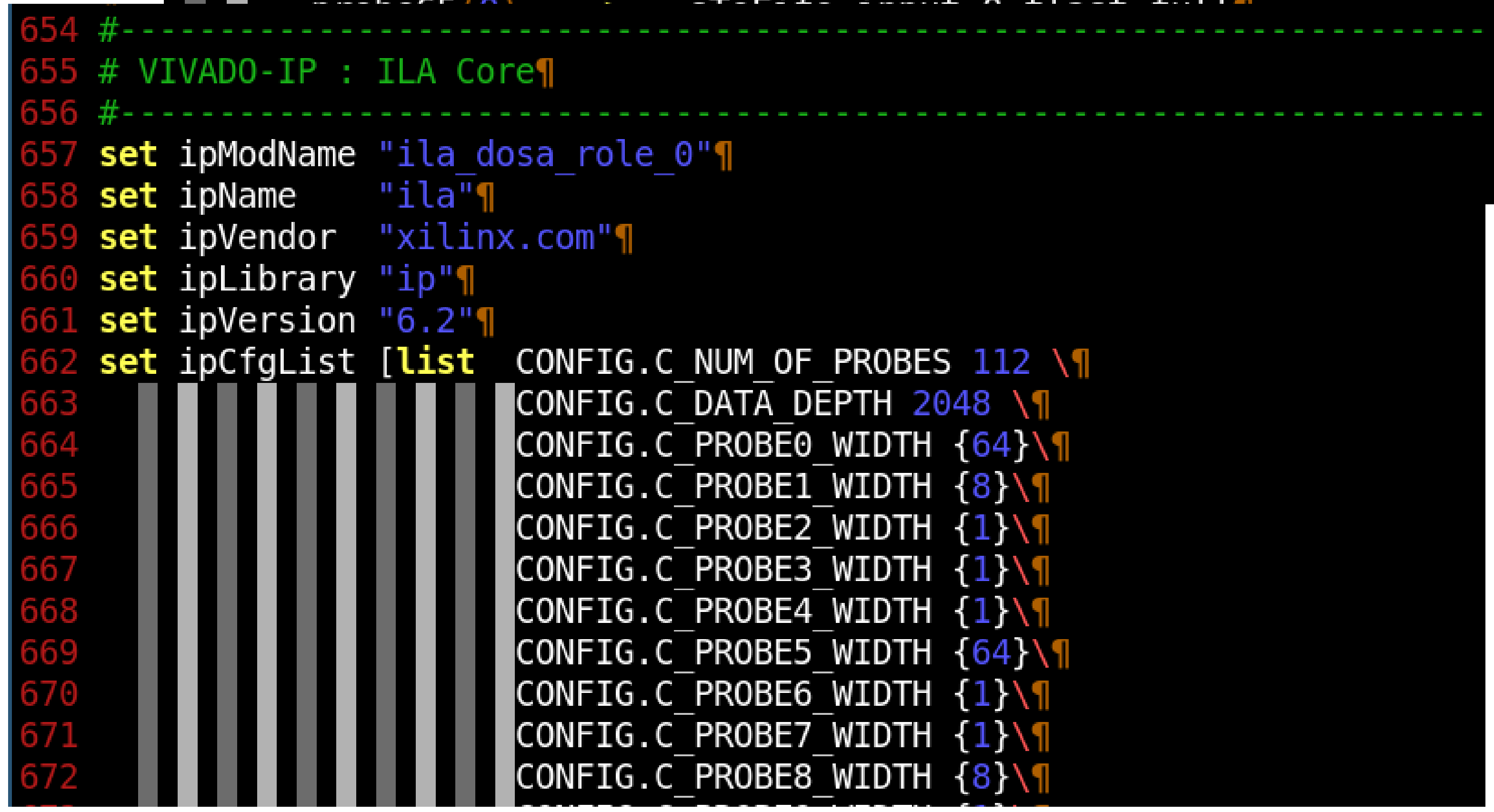


Debugging generated by compiler

- Distributed inference means distributed debugging
  → compiler must facilitate it
- Hence, DOSA automatically generates debug probes between IP cores
  - Because we use standardized interfaces between IP cores → easily generate able by compiler
  - In VHDL and tcl
- We deploy bitstreams using partial reconfiguration → debug bridge support
- (...then we still have to look at waveforms...)



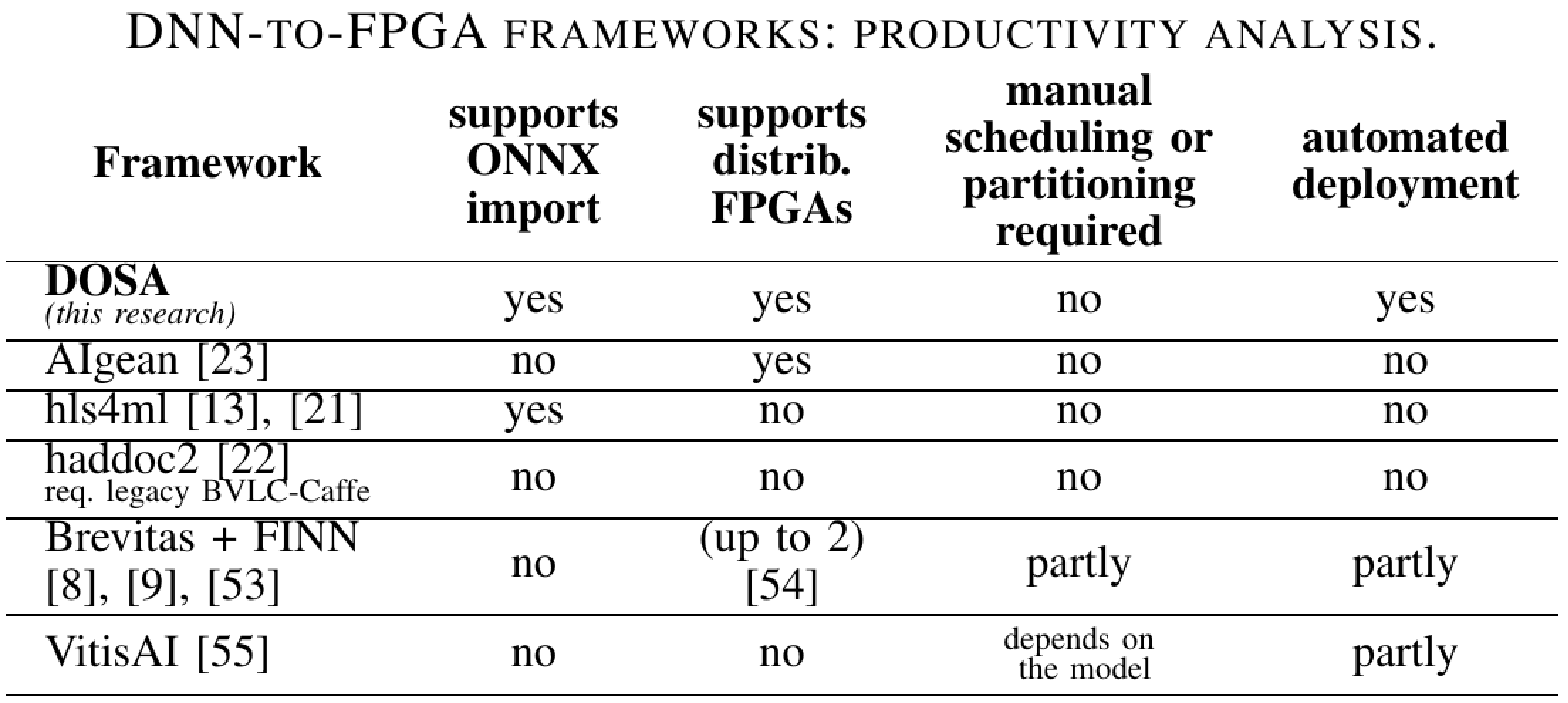
```
--##########################################################
--  Debug Core instantiation¶
--##########################################################

DBG: ila_dosa_role_0¶
  port map (¶
    clk => piSHL_156_25Clk¶
    , probe0      =>   siNRC_Udp_Data_tdata¶
    , probe1      =>   siNRC_Udp_Data_tkeep¶
    , probe2(0)   =>   siNRC_Udp_Data_tvalid¶
    , probe3(0)   =>   siNRC_Udp_Data_tlast¶
    , probe4(0)   =>   siNRC_Udp_Data_tready¶
    -- ...¶
    , probe52      =>   sMPE_Debug¶
    , probe53      =>   sZRLMPI_Wrapper_Debug¶
    , probe54(0)   =>   sResetApps_n¶
    , probe55      =>   sToFifo_input_0_tdata_din¶
    , probe56(0)   =>   sToFifo_input_0_tdata_full_n¶
    , probe57(0)   =>   sToFifo_input_0_tdata_full¶
    , probe58(0)   =>   sToFifo_input_0_tdata_write¶
    , probe59      =>   sToFifo_input_0_tkeep_din¶
    , probe60(0)   =>   sToFifo_input_0_tkeep_full_n¶
    , probe61(0)   =>   sToFifo_input_0_tkeep_full¶
    , probe62(0)   =>   sToFifo_input_0_tkeep_write¶
    , probe63      =>   sToFifo_input_0_tlast_din¶
    , probe64(0)   =>   sToFifo_input_0_tlast_full_n¶
```



```
#------------------------------------------------------------------
# VIVADO-IP : ILA Core¶
#------------------------------------------------------------------
set ipModName "ila_dosa_role_0"¶
set ipName    "ila"¶
set ipVendor  "xilinx.com"¶
set ipLibrary "ip"¶
set ipVersion "6.2"¶
set ipCfgList [list  CONFIG.C_NUM_OF_PROBES 112 \¶
                     CONFIG.C_DATA_DEPTH 2048 \¶
                     CONFIG.C_PROBE0_WIDTH {64}\¶
                     CONFIG.C_PROBE1_WIDTH {8}\¶
                     CONFIG.C_PROBE2_WIDTH {1}\¶
                     CONFIG.C_PROBE3_WIDTH {1}\¶
                     CONFIG.C_PROBE4_WIDTH {1}\¶
                     CONFIG.C_PROBE5_WIDTH {64}\¶
                     CONFIG.C_PROBE6_WIDTH {1}\¶
                     CONFIG.C_PROBE7_WIDTH {1}\¶
                     CONFIG.C_PROBE8_WIDTH {8}\¶
```

Gains? → Evaluation

- Main problem is not technical unfit of DNN-to-FPGA frameworks, but their usability
- DOSA tries to change that with:
  - Support of **major community standards** (foremost ONNX)
  - **No architectural knowledge necessary** at the user side
  - Automated **partitioning**
  - Automated **deployment**

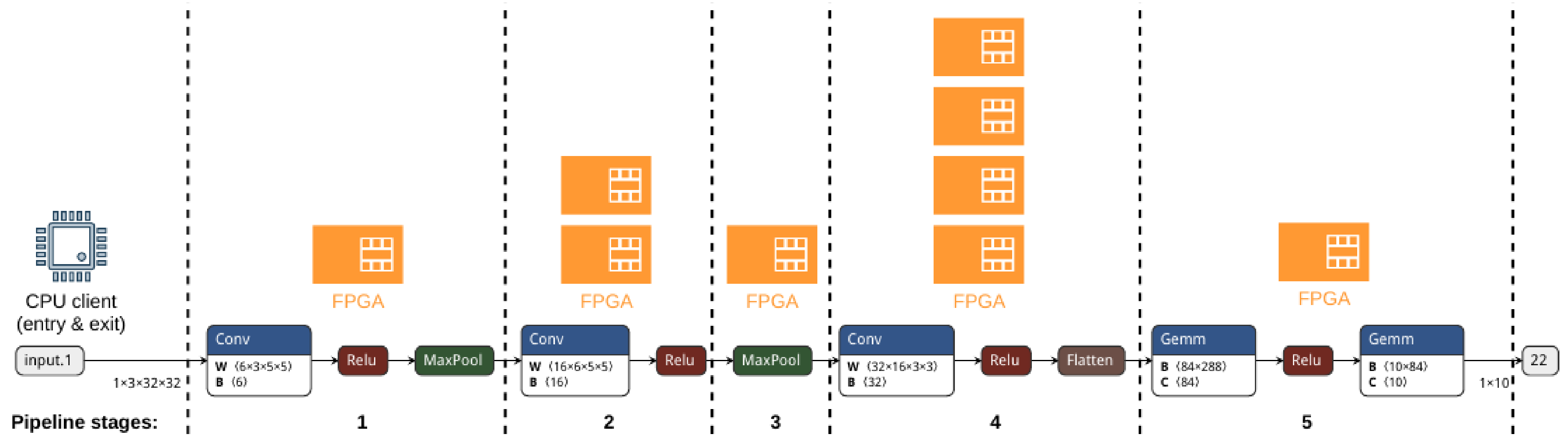


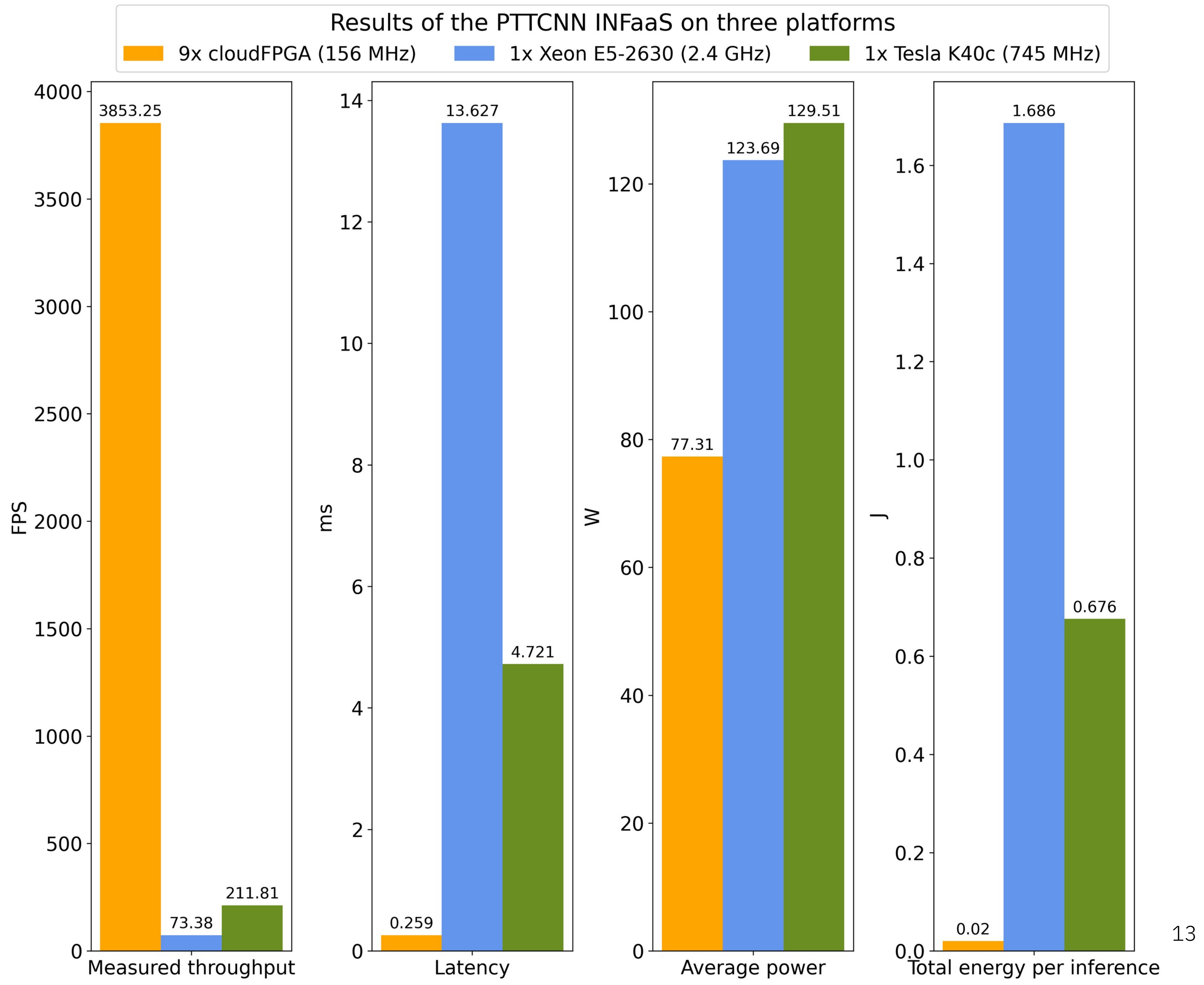
DNN-TO-FPGA FRAMEWORKS: PRODUCTIVITY ANALYSIS.

Framework	**supports ONNX import**	**supports distrib. FPGAs**	**manual scheduling or partitioning required**	**automated deployment**
DOSA *(this research)*	yes	yes	no	yes
AIgean [23]	no	yes	no	no
hls4ml [13], [21]	yes	no	no	no
haddoc2 [22] req. legacy BVLC-Caffe	no	no	no	no
Brevitas + FINN [8], [9], [53]	no	(up to 2) [54]	partly	partly
VitisAI [55]	no	no	depends on the model	partly

DNN-to-FPGAs: some results

- DOSA allows a “one-click” design-space exploration, partitioning, compilation, and synthesis
  - Deployment automated with ZRLMPI
  - Optimization within seconds (not hours)
- E.g. small CNN for INFaaS (i.e. batch size 1) across 9 cloudFPGAs results in (8 bit weights):
  - **>50x speedup, >80x more efficient vs. CPU**
  - **>18x speedup, >30x more efficient vs. GPUs**
  - End-to-end latency below 0.3ms (client-side)
- Speedup and efficiency gains of cloudFPGA:
  - massive parallelism (streaming-architecture) and custom data paths
  - direct network-attachment
  - disaggregation



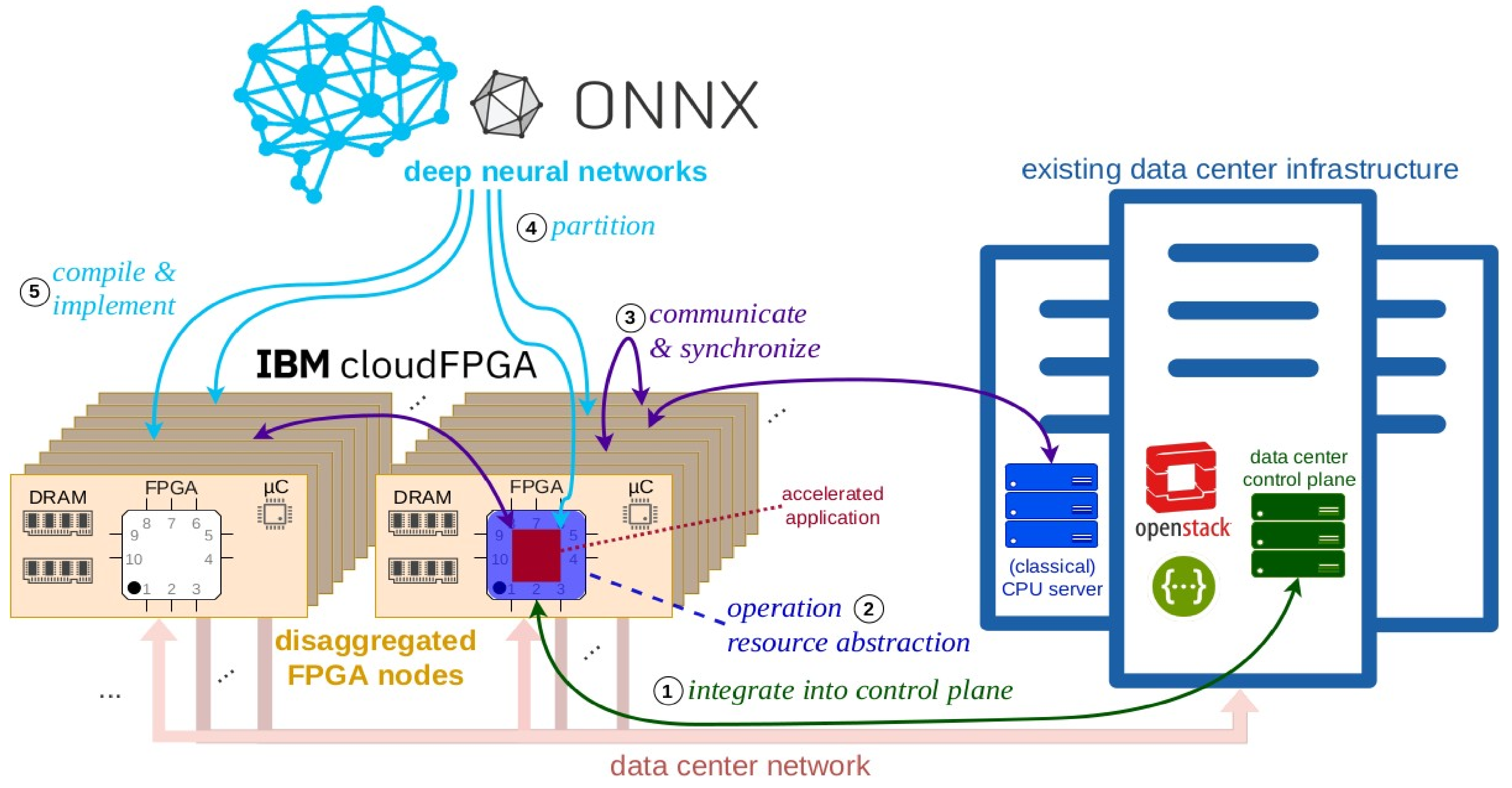


Conclusion?

- To boost adoption of FPGAs → holistic approach with organic compilers
  - Wide range of DNNs
  - Usable by non-FPGA experts
- Operation Set Architecture overcome current hurdles of DNN-to-FPGA frameworks
- DOSA: One-click open-source → **github.com/cloudFPGA/DOSA**
  - Increase scope of potential solutions
  - Automatic distribution across FPGAs

➔ Efficient automated distributed DNN inference:
  - >50x speedup, >80x more efficient vs. CPU
  - >18x speedup, >30x more efficient vs. GPUs

➔ An "FPGA standard stack" must be open source!



Devices	Measured throughput (fps)	Latency (ms)	Average Power (W)	Total energy per inference (J)
9x KU060 FPGAs (156 MHz)	3,853.25*	0.259	77.31	0.020
1x Xeon E5-2630 (2.4 GHz)	73.38	13.627	123.69	1.686
1x Tesla K40c (745 MHz)	211.81	4.721	129.51	0.676

*: Limited by the single-threaded CPU client.

...looking forward to all your questions!

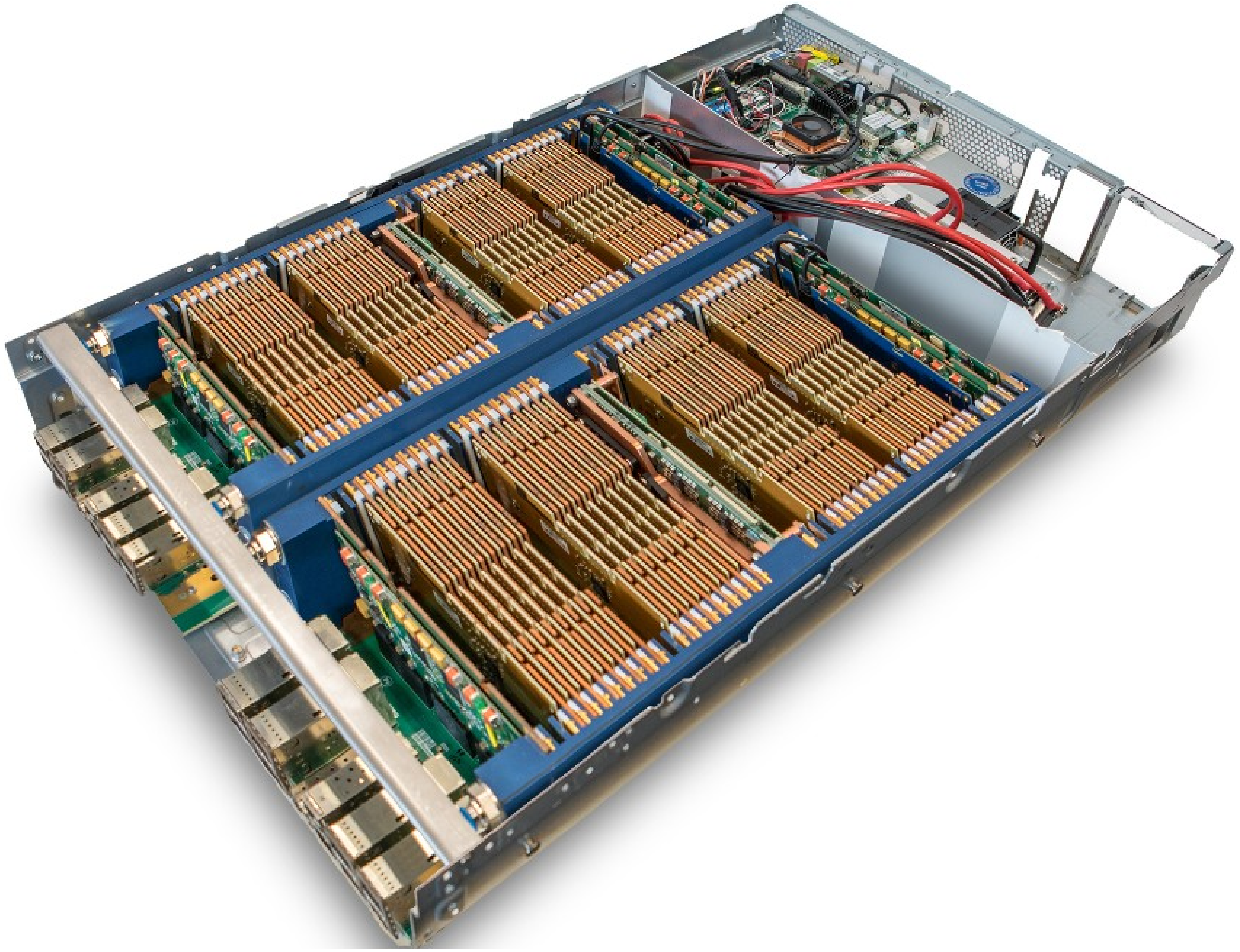
Dr. Burkhard Ringlein
ngl@zurich.ibm.com
research.ibm.com/projects/cloudfpga
burkhard-ringlein

Appendix

As testbed for distributed edge FPGA environments: The IBM cloudFPGA Platform

- 19”x2U w/64 FPGAs
- Network-attached, disaggregated FPGAs
- 640GbE fully balanced

(more information at github.com/cloudfpga)



References

Sources are referenced in the slides directly.

All remaining images are from IBM DAM or IBM Websites or created by the author or the EVEREST consortium. © 2023 IBM Corporation.

IBM cloudFPGA: Further Reading

- https://github.com/cloudFPGA
- The **cloudFPGA project page at ZRL:** https://www.zurich.ibm.com/cci/cloudFPGA/
- B. Ringlein, F. Abel, D. Diamantopoulos, B. Weiss, C. Hagleitner, and D. Fey, “Advancing Compilation of DNNs for FPGAs using Operation Set Architectures,” in IEEE Computer Architecture Letters, 2022.
- B. Ringlein, F. Abel, D. Diamantopoulos, B. Weiss, C. Hagleitner, M. Reichenbach and D. Fey, “A Case for Function-as-a-Service with Disaggregated FPGAs,” in Proceedings of the 2021 IEEE 14th International Conference on Cloud Computing (CLOUD).
- B. Ringlein, F. Abel, A. Ditter, B. Weiss, C. Hagleitner and D. Fey, “Programming Reconfigurable Heterogeneous Computing Clusters Using MPI With Transpilation”, 2020 IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC).
- B. Ringlein, F. Abel, A. Ditter, B. Weiss, C. Hagleitner and D. Fey, “ZRLMPI: A Unified Programming Model for Reconfigurable Heterogeneous Computing Clusters” in 28th IEEE International Symposium On Field-Programmable Custom Computing Machines (FCCM), 2020.
- B. Ringlein, F. Abel, A. Ditter, B. Weiss, C. Hagleitner and D. Fey, “ System architecture for network-attached FPGAs in the cloud using partial reconfiguration,” in 29th International Conference on Field Programmable Logic and Applications (FPL), 2019.
- F. Abel, J. Weerasinghe, C. Hagleitner, B. Weiss, S. Paredes, “An FPGA Platform for Hyperscalers,” in IEEE 25th Annual Symposium on High-Performance Interconnects (HOTI), Santa Clara, CA, pp. 29–32, 2017.
- F. Abel, “How do you squeeze 1000 FPGAs into a DC rack?” online at LinkedIn

Notices