Demonstrating NADA

A Workflow for Distributed CNN Training on FPGA Clusters

Philipp Kreowsky[*†], Justin Knapheide[*†], Benno Stabernack[*†]

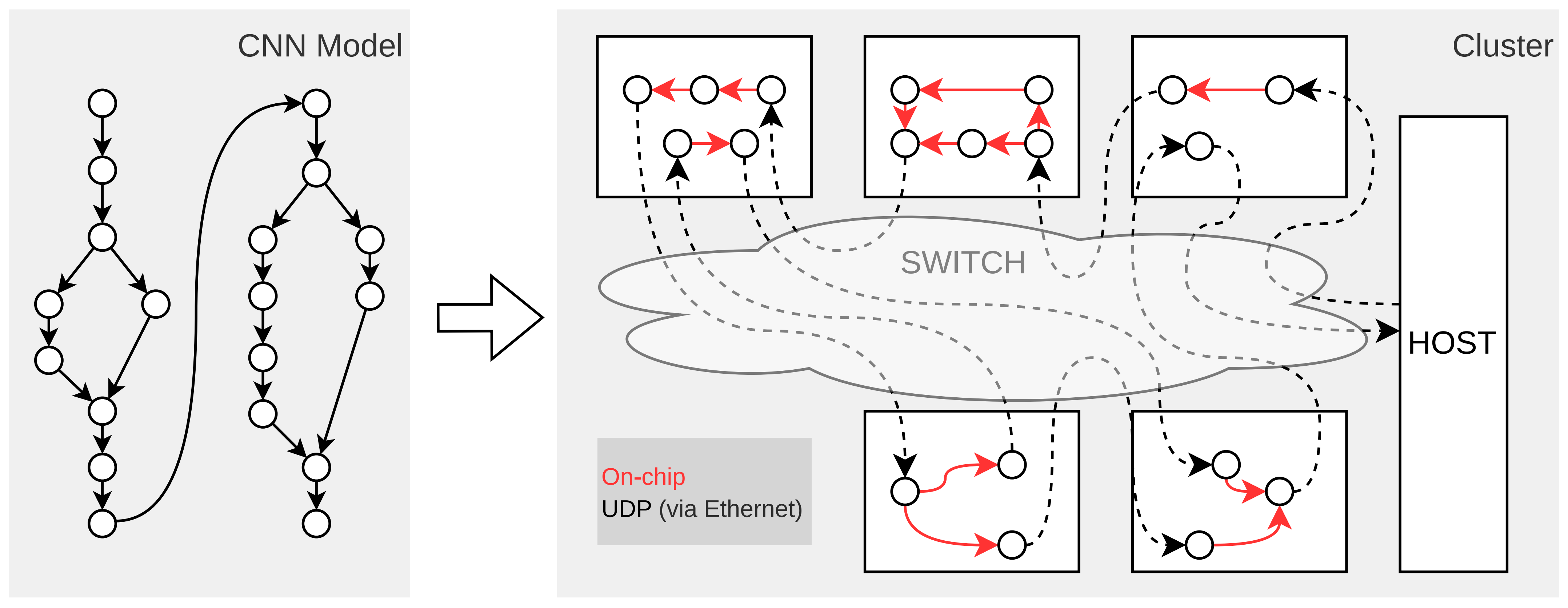
[*]Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Berlin, Germany
{philipp.kreowsky, justin.knapheide, benno.stabernack}@hhi.fraunhofer.de
[†]University of Potsdam, Embedded Systems Architectures for Signal Processing, Potsdam, Germany

Abstract

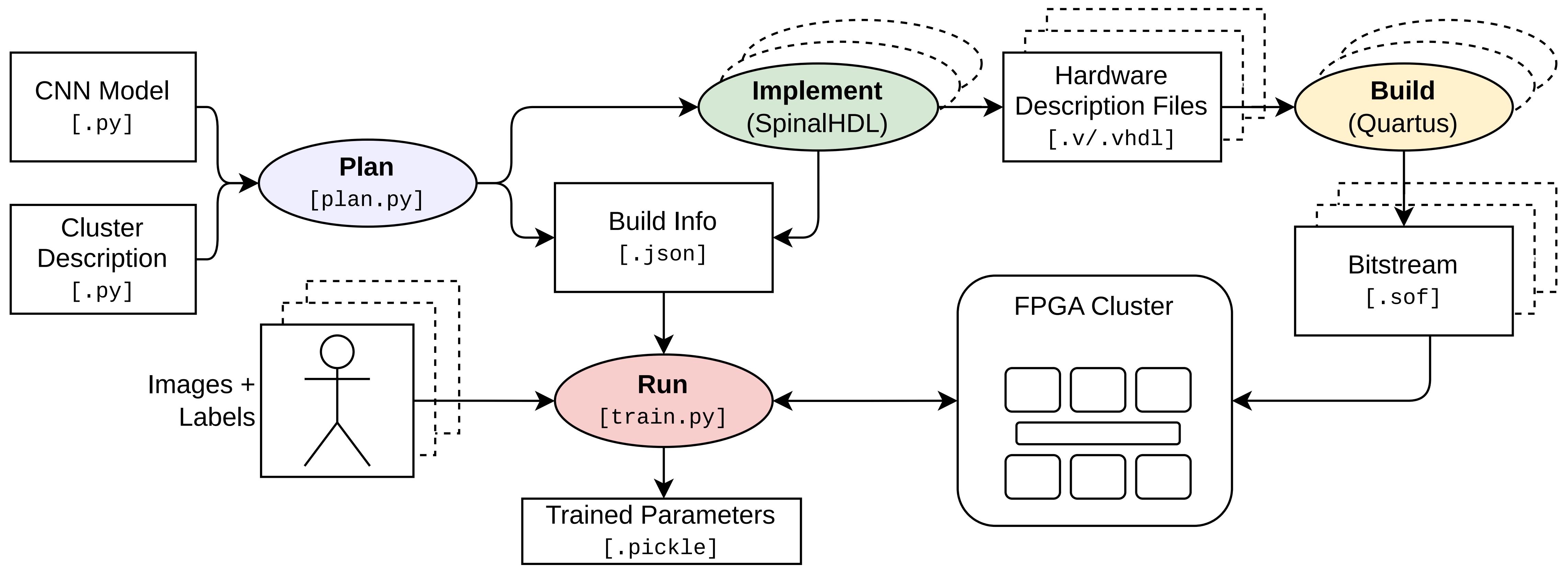
Network **A**ttached **D**eep learning **A**ccelerator

- **Training** of deep neural networks on FPGA clusters
- Flexible HW/SW framework
- Arbitrary **DAG** structure
- Focus on **layer parallelism**
- Dedicated implementation for each layer
- **Automatic** placement across FPGAs in the cluster
- Tightly pipelined
- Inter-device communication via **UDP/IP**
- Implemented in **SpinalHDL**
- Resource estimates derived from hardware description

Layer-Parallel Mapping of CNN to Cluster



Tool Flow



Build

- **Common interface** for implemented devices
- Device-specific toplevel
- Device-specific arithmetic IP cores
- Intel **Quartus** for synthesis and bitstream generation

Run

1. **Write bitstreams** to FPGAs in the cluster
2. **Set up** inter-FPGA **connections** according to **Build Info**
3. **Write** batch of **images** via UDP/IP, throttled to maximum throughput
4. **Wait** for completion
5. Trigger **parameter update** (SGD)
6. Repeat from 3

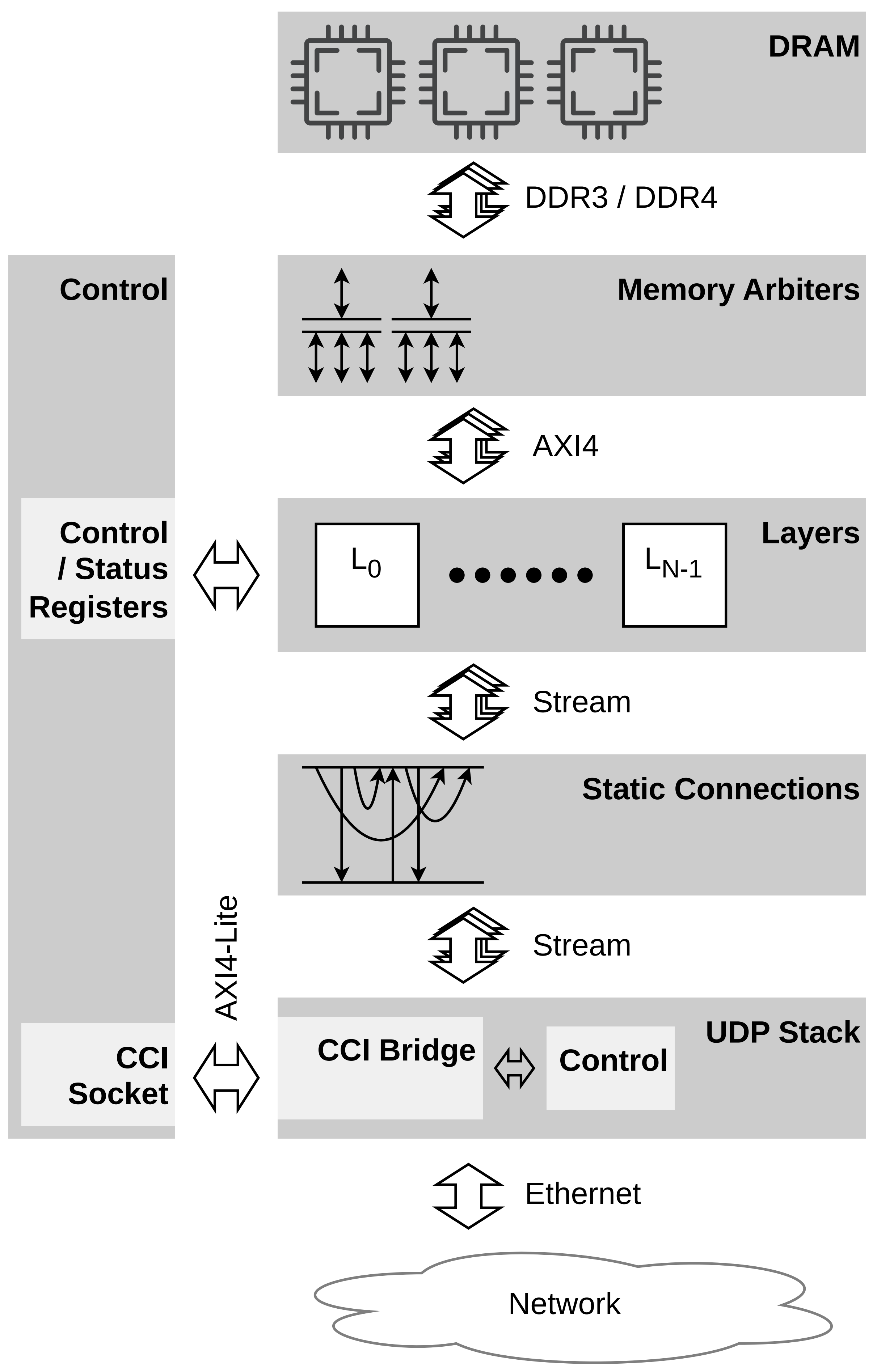
Plan

- Input: **Model**, **Cluster**
- Output: **Mapping** of layers to FPGAs, **generics** for layer implementations
- Model description is DAG of supported layers
- Cluster description lists available amounts of relevant resources
  - **on-chip memory** – Register, MLAB, M20K
  - **DRAM**
  - **DSPs**
- Dedicated implementation for each layer
  - Resource requirements extracted from early run of **Implement** step
- **Layer-parallel** $\Rightarrow$ Overall throughput limited by the slowest layer implementation
- Given the available resources, maximize the throughput
- For fixed target throughput, becomes a **constraint satisfaction problem**
- Binary search for optimal throughput using a SAT solver

Implement

- Using **SpinalHDL**
- **Common interface** for layer implementations
- **Streams** for data transfer
  - Features `Seq[Stream[Vec[Floating]]]`
  - Feature gradients `Seq[Stream[Vec[Floating]]]`
  - Parameters `Stream[Floating]`
- Layer implementation instantiated and connected according to **Plan** step
- **UDP/IP** on top of **40 GbE** / **100 GbE** for inter-FPGA communication

Implemented Architecture on each FPGA



Execution Times

Model	Cluster	Plan	Build	Run	FPS (measured / planned)
MobileNetV2	4 × AGFB014	21 min	244 min	160 h	332 / 500
Demo	4 × 10AX115	3 min	218 min	140 h	378 / 558
VGG mini	2 × 10AX115	7 min	156 min	99 h	532 / 600

Demo Night FPL 2023, Gothenburg