Pacific Northwest NATIONAL LABORATORY

DATE 26

Agile Hardware Design for Complex Data Science Applications: Opportunities and Challenges

April 28, 2026

Antonino Tumeo

Chief Scientist

U.S. DEPARTMENT of ENERGY BATTELLE

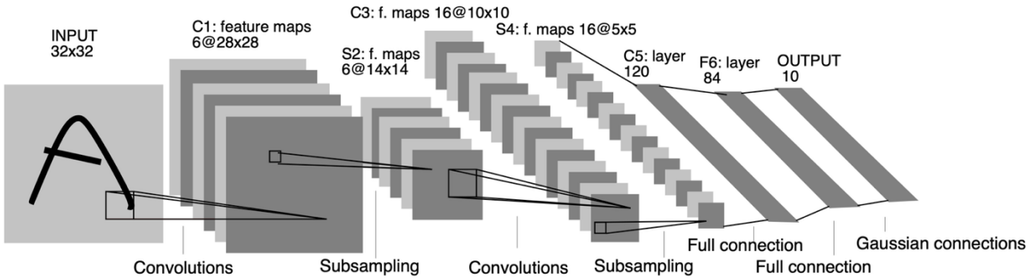
PNNL is operated by Battelle for the U.S. Department of Energy

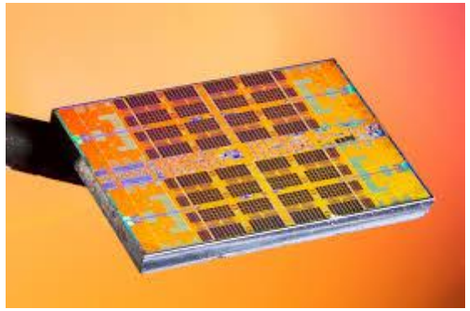
Motivations

- Data science algorithms, approaches, and frameworks are quickly evolving
- Domain-specific accelerators are the only possible approach to keep increasing performance in tight constraints
- Existing accelerators start from specific models (i.e., mostly deep neural networks) or only try to accelerate specific computational patterns coming from high-level frameworks
- Designing hardware by hand is complex and time-consuming
- Depending on the application, a designer may want to explore performance, area, energy, accuracy, and more…

- ***Need tools to quickly transition from formulation of an algorithm to the accelerator implementation and explore the accelerator design along different dimensions***

LeNet architecture from the original paper





Pacific Northwest
NATIONAL LABORATORY

Why Data Science?

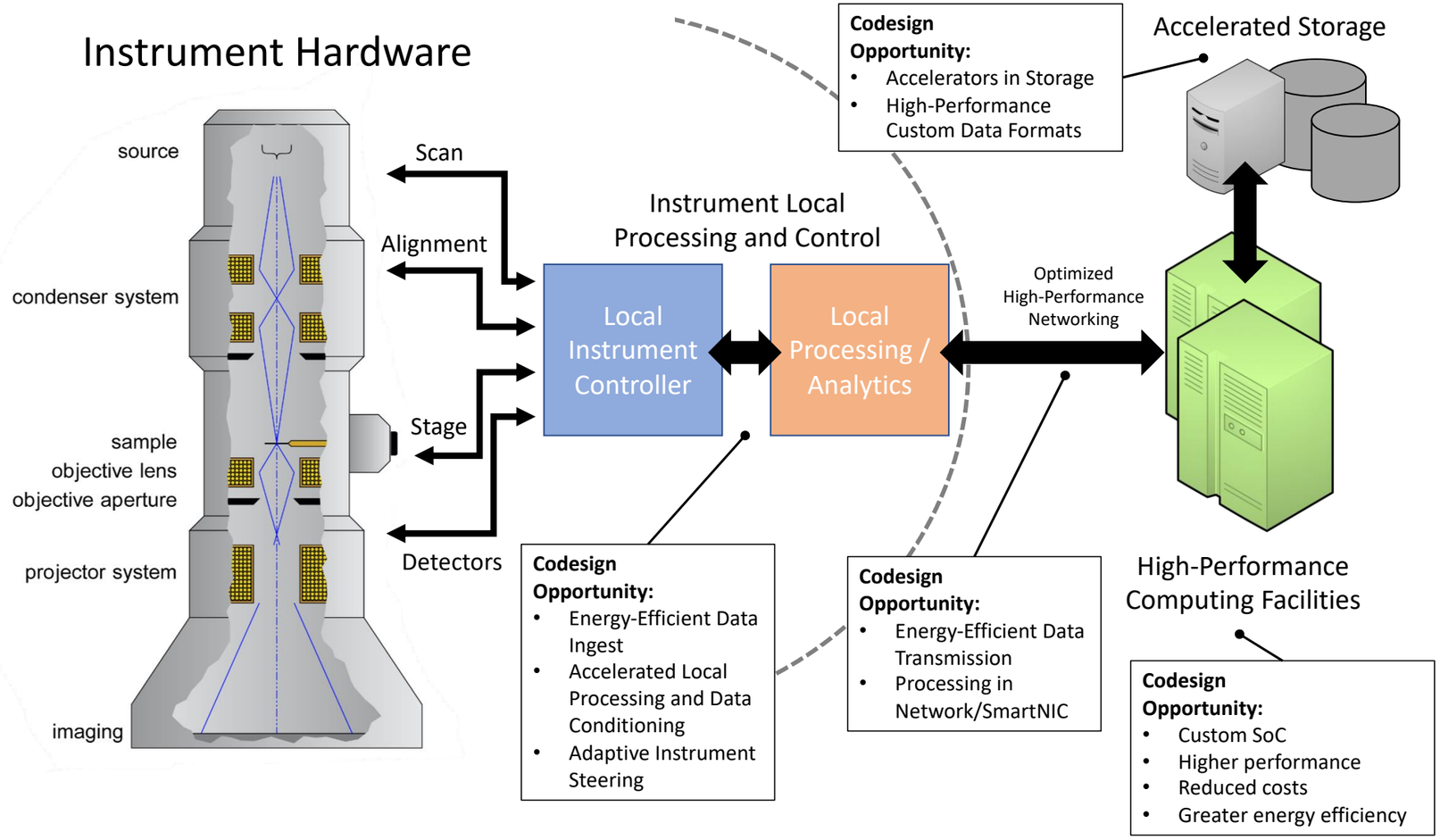
- Increasingly complex data analysis pipelines
- May include algorithms with significantly different behaviors
  - Deep neural networks, graph analytics, graph representation learning…
- Algorithmic research in the area is quickly evolving
- Algorithms are data-intensive
  - Significant amount of data per computation
- Some algorithms exhibit irregular behaviors
  - Graph algorithms are the prototypical irregular kernel

Pacific Northwest NATIONAL LABORATORY

Possible Applications

- Inference in the cloud (Brainwave, Bing, Alibaba, Amazon…)
- High-performance computing
  - Converged applications (Scientific simulation, machine learning, and graph analytics)
  - Near data / near network data analysis
- Autonomous systems
  - Low latency reasoning for decision making
  - Federated learning

Possible Applications: Experimental Scientific Workflows

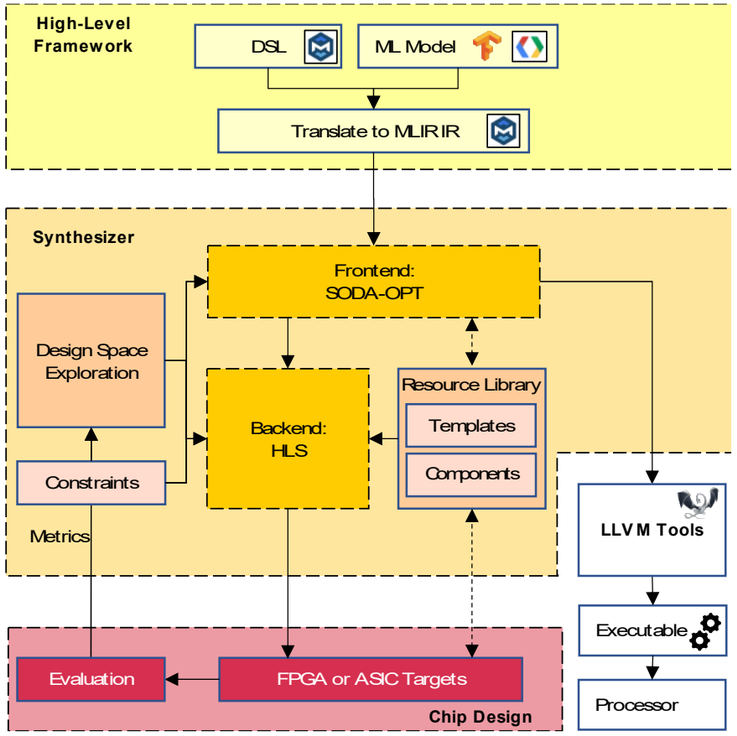


Pacific Northwest
NATIONAL LABORATORY

Challenges

- Need to go from the high-level data science frameworks to the hardware implementation
  - Tension between domain specificity and generality applies to both hardware and the hardware generators/tools
  - Python frameworks typically based on tensor representations
    - ✓ What about graphs and sparse data structures?
- Generating only the accelerators is not sufficient, we need to consider the system level implications
- Many levels, many tools, often not directly interoperating
- Verification and testing
- Many efforts to reduce costs to access tools and IPs, but still a long road

SODA Synthesizer: Overview



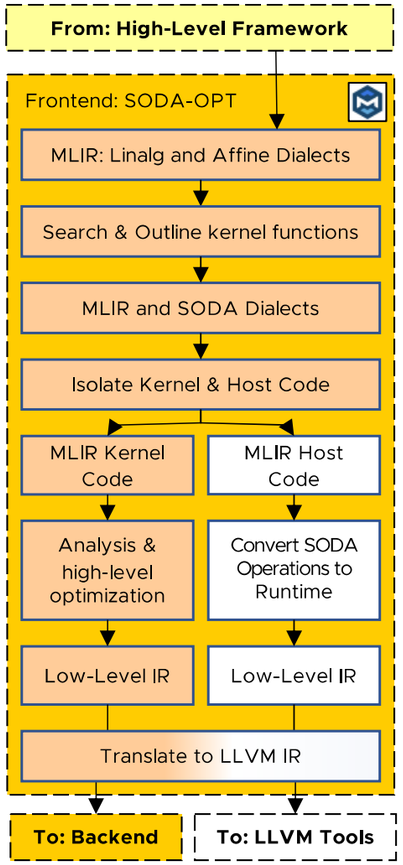
[M. Minutoli, V. G. Castellana, C. Tan, J. Manzano, V. Amatya, A. Tumeo, D. Brooks, G-Y. Wei: SODA: a New Synthesis Infrastructure for Agile Hardware Design of Machine Learning Accelerators. ICCAD 2020: 98:1-98:7]

[J. Zhang, N. Bohm Agostini, S. Song, C. Tan, A. Limaye, V. Amatya, J. Manzano, M. Minutoli, V. G. Castellana, A. Tumeo, G-Y. Wei, D. Brooks: Towards Automatic and Agile AI/ML Accelerator Design with End-to-End Synthesis. ASAP 2021: 218-225]

- A modular, multi-level, interoperable, extensible, **open-source hardware compiler** from **high-level programming frameworks to silicon**
- Compiler-based frontend, leveraging the MultiLevel Intermediate Representation (MLIR)
- **Compiler-based backend**, leveraging state-of-the-art High-Level Synthesis (HLS) techniques, as well as a Coarse-Grained Reconfigurable Array (CGRA) generator
- Generates **synthesizable Verilog** for a variety of targets, from Field Programmable Gate Arrays (FPGAs) to Application Specific Integrated Circuits (ASICs)
- Optimizations at all levels are performed as **compiler optimization** passes

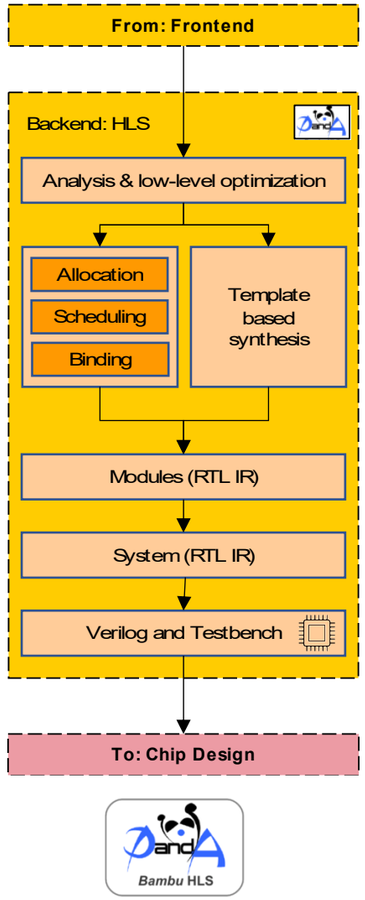
[N. Bohm Agostini, S. Curzel, J. Zhang, A. Limaye, C. Tan, V. Amatya, M. Minutoli, V.G. Castellana, J. Manzano, A. Tumeo: Bridging Python to Silicon: The SODA Toolchain. IEEE Micro Magazine 2022]
[N. Bohm Agostini, S. Curzel, V. Amatya, C. Tan, M. Minutoli, V. G. Castellana, J. Manzano, D. Kaeli, A. Tumeo : An MLIR-based Compiler Flow for System-Level Design and Hardware Acceleration. ICCAD 22]
[Fabrizio Ferrandi, Vito Giovanni Castellana, Serena Curzel, Pietro Fezzardi, Michele Fiorito, Marco Lattuada, Marco Minutoli, Christian Pilato, Antonino Tumeo: Invited: Bambu: an Open-Source Research Framework for the High-Level Synthesis of Complex Applications. DAC 2021: 1327-1330]

SODA Synthesizer: Components

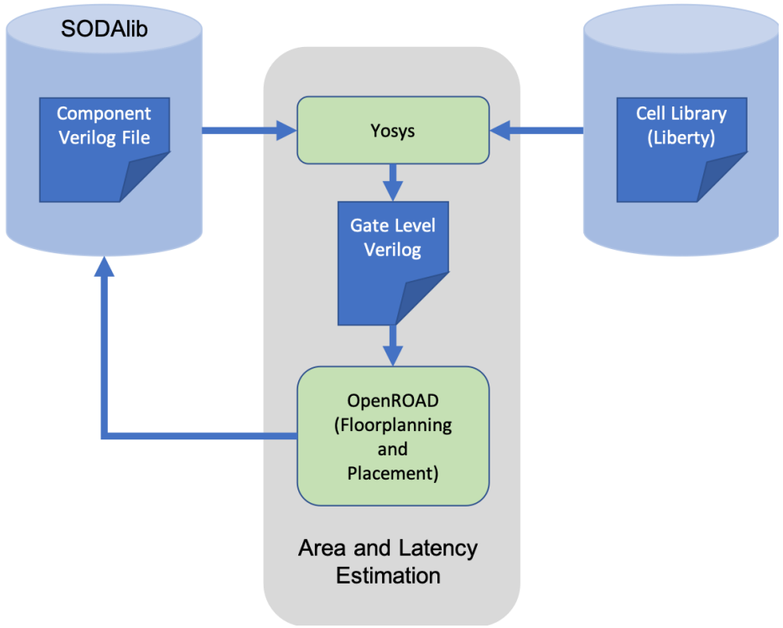


SODA-OPT

https://github.com/pnnl/soda-opt



https://panda.dei.polimi.it

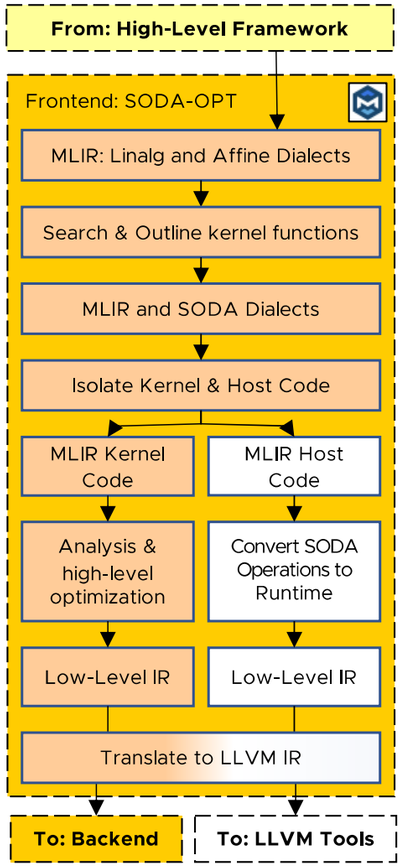


OpenROAD

https://theopenroadproject.org

SODA-OPT: Frontend and High-Level IR

- **SODA-OPT: S**earch, **O**utline, **D**ispatch, **A**ccelerate frontend **opt**imizer “generates” the SODA High-Level IR
- Employs and embraces the MLIR framework
  - MLIR: Multi-Level Intermediate Representation
  - Used in TensorFlow, TFRT, ONNX-MLIR, NPComp, others
  - Several architecture independent dialects (Linalg, Affine, SCF) and optimizations
- Interfaces with high-level ML frameworks through MLIR “bridges” (e.g., libraries, rewriters)
- Defines the SODA MLIR dialect and related compiler passes to:
  - Identify dataflow segments for hardware generation
  - Perform high-level optimizations (dataflow transformations, data-level and instruction-level parallelism extraction)
  - Generate interfacing code and runtime calls for microcontroller



SODA-OPT: System Overview

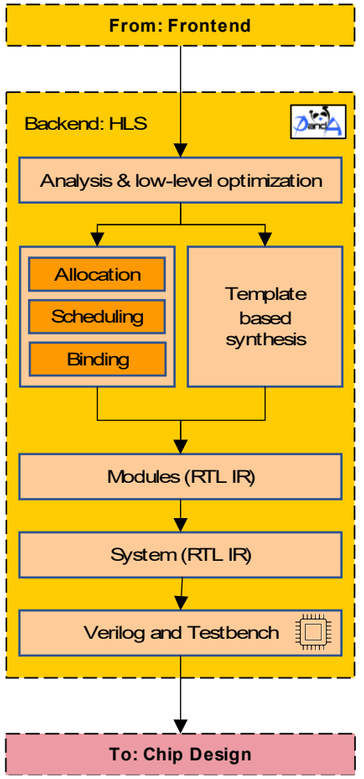
https://github.com/pnnl/soda-opt

[N. Bohm Agostini, S. Curzel, V. Amatya, C. Tan, M. Minutoli, V. G. Castellana, J. Manzano, D. Kaeli, A. Tumeo : An MLIR-based Compiler Flow for System-Level Design and Hardware Acceleration. ICCAD 22]

[N. Bohm Agostini, S. Curzel, D. Kaeli, A. Tumeo: SODA-OPT an MLIR based flow for co-design and high-level synthesis. CF 2022: 201-202 - **Best Poster Award**.]

SODA Synthesizer: HLS Backend

- The synthesizer backend take as input the properly optimized low-level IR and generate the hardware descriptions of the accelerators
- The HLS backend is PandA-Bambu, an open-source state-state-of-the-art High-Level Synthesis (HLS) framework
  - Key features: **parallel accelerator designs**, **modular HLS**, and **ASIC support**
- The HLS backend provides automated testing and verification of the generated designs



Bambu HLS

https://panda.dei.polimi.it

[Fabrizio Ferrandi, Vito Giovanni Castellana, Serena Curzel, Pietro Fezzardi, Michele Fiorito, Marco Lattuada, Marco Minutoli, Christian Pilato, Antonino Tumeo: Invited: Bambu: an Open-Source Research Framework for the High-Level Synthesis of Complex Applications. DAC 2021: 1327-1330]

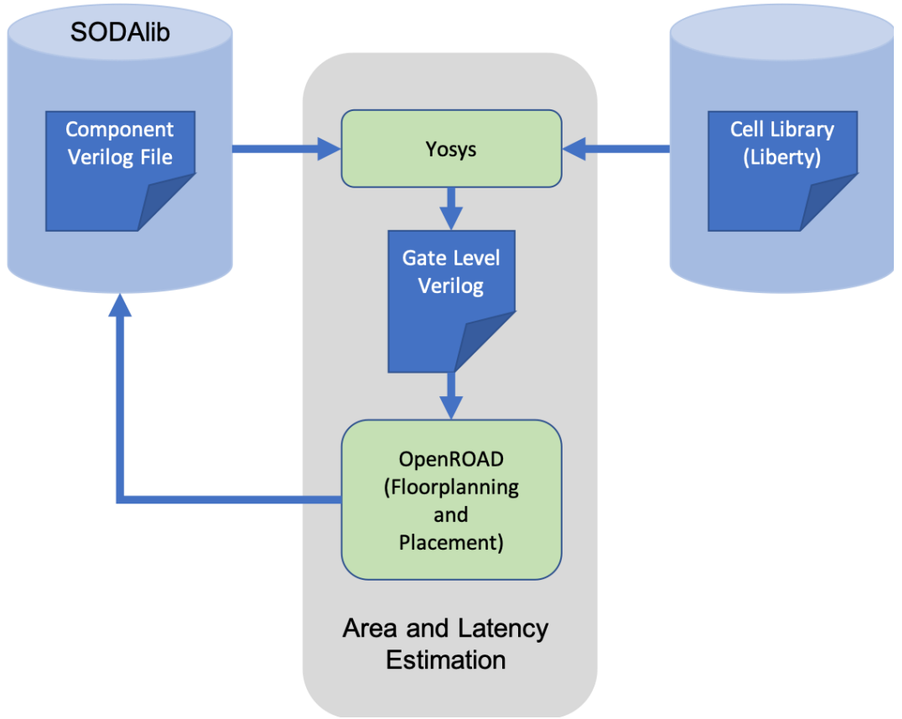
SODA Synthesizer: ASIC targets

- The multi-level approach of the SODA toolchain allows supporting different target technologies (FPGA, ASIC) for actual generation of the designs

- SODA also supports ASIC targets:
  - **Commercial Tools** (Synopsys Design Compiler with Global Foundries 12/14 nm cells)
  - **OpenROAD suite** (OpenPDK 45nm and ASAP 7nm cell libraries)

- Backends' resources characterized for the target technology:
  - **HLS Backend: Eucalyptus** tool in Bambu, allows driving hardware synthesis algorithms to optimize for area, latency, etc.

- PandA-Bambu now also interfaces with the opensource C frontend for **ZeroASIC' SIliconCompiler** (https://www.siliconcompiler.com)



SODA characterization flow. The characterization flow can be extended to synthesize HLS generated designs, or used to estimate their area-latency-power profiles to drive the Design Space Exploration engine

OpenROAD

https://theopenroadproject.org

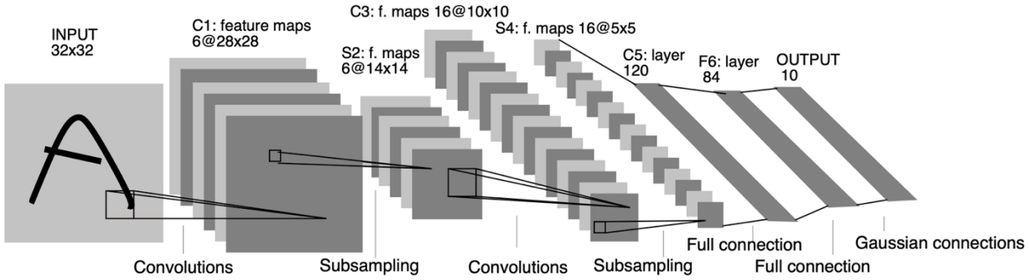
Pacific Northwest
NATIONAL LABORATORY

Why an Open-Source High-Level Synthesis Toolchain to enable Scientific Computing?

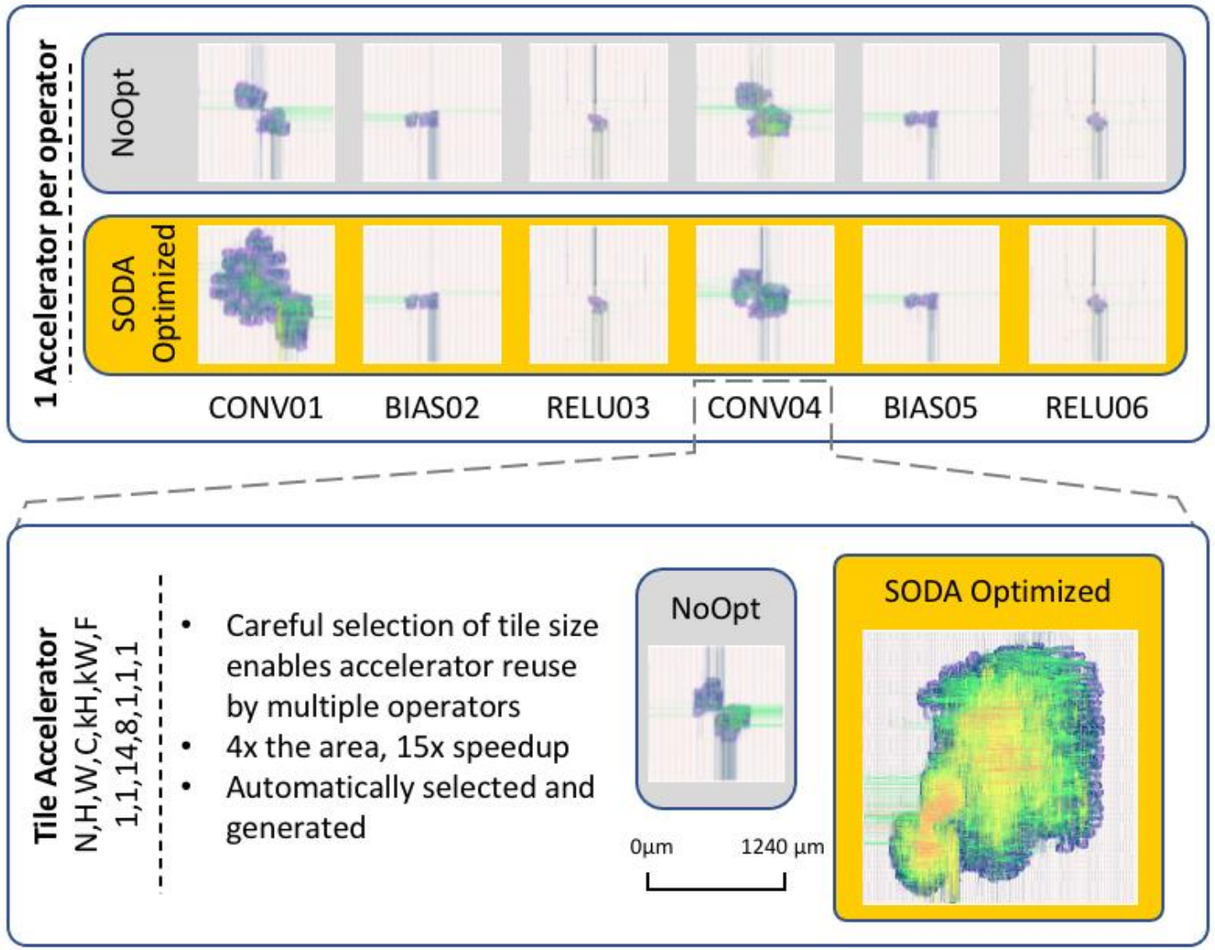
- Scientific computing workflows are evolving rapidly and becoming more complex
  - Moving towards autonomous science on a continuum of computing systems
- The continuum of computing is, by definition, highly heterogeneous, with contrasting requirements
  - Domain-specific accelerators are the only way forward
- Conventional HLS tools leverage C/C++ annotated with hardware and tool specific pragmas
  - Developer still need to do significant optimizations by hand, the productivity gap for writing hardware description languages becomes a productivity gap for writing highly specialized C/C++
- We need tools that democratize hardware design **AND** allow to perform research of new methodologies to address new applications and platforms

From Python to optimized ASIC

LeNet architecture from the original paper

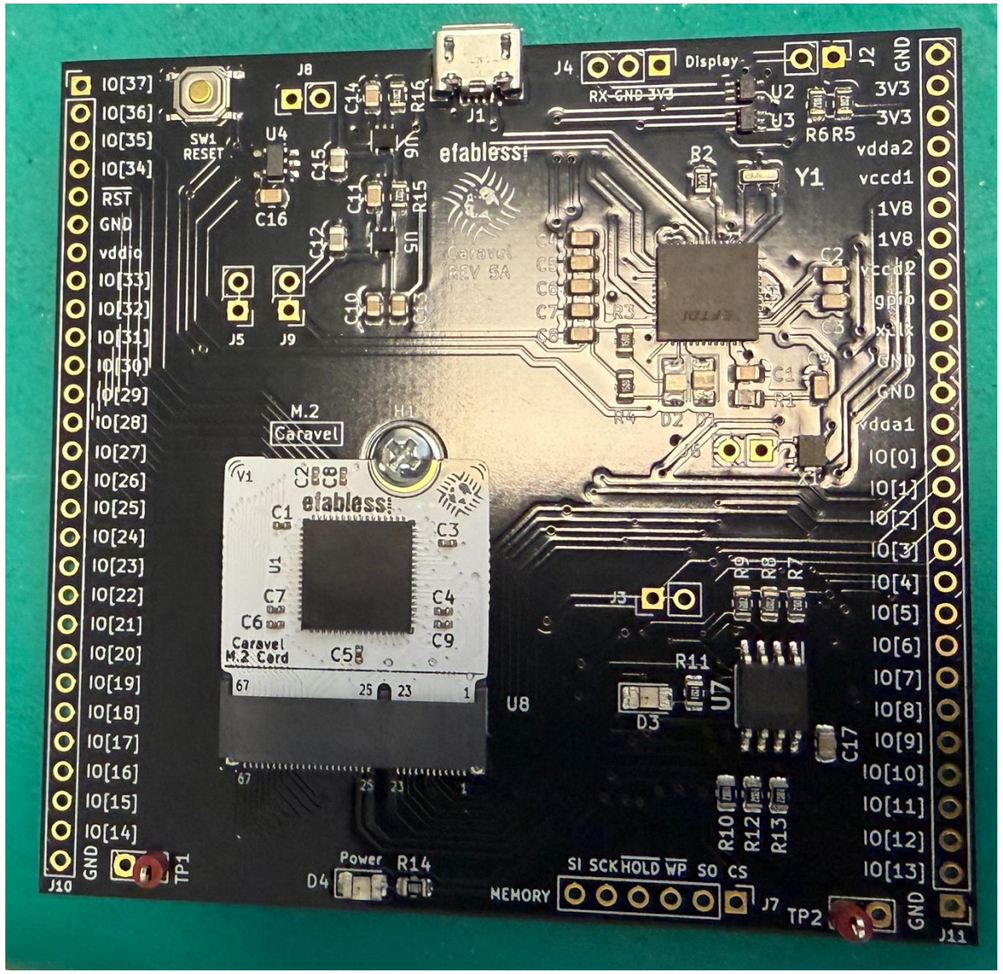


- LeNet example
- Each of the operator is synthesized to an **ASIC accelerator**
- SODA optimized accelerators are bigger, but also much **faster**



Literally, from Python to Silicon, only with Open-source Tools

- eFabless flow (now Chipfoundry)
- Uses SkyWater 130 nm open process development kit (PDK)
- Uses OpenLane (repackaged OpenROAD flow) for physical Design and layout
- Implements exactly the baseline, unoptimized accelerator of the SODA tutorial
  - Only modification: insert SRAM macro between harness platform microcontroller (PicoRV) and wishbone
  - Bambu can generate also wishbone interfaces



Pacific Northwest NATIONAL LABORATORY

Other Approaches (an incomplete list)

- A multitude of hand-designed domain specific accelerators
  - Trying to recover some levels of flexibility either using extensible Instruction Set Architectures (e.g., RISC-V) or novel reconfigurable designs
- Approaches that generate custom hardware from Python framework mapping on parametric templates
  - GEMMINI: parametric template
  - VeriGOOD-ML: compiler maps on three different architectures
  - VTA: specialized coprocessor (GEMM unit) generated with HLS
- Convert code to imperative languages (C/C++) annotated for HLS
  - PyLog: Python to C/C++ for Vivado HLS
  - HeteroCL: partitions code between CPU and FPGA, provides a library of functions to insert hardware-specific information in the source code, generates C/C++ for HLS
  - ScaleHLS: MLIR to annotate C/C++ for Vivado HLS
- CIRCT (Circuit IR Compilers and Tools): MLIR to build interoperable tools for hardware design

Other Approaches (an incomplete list continued)

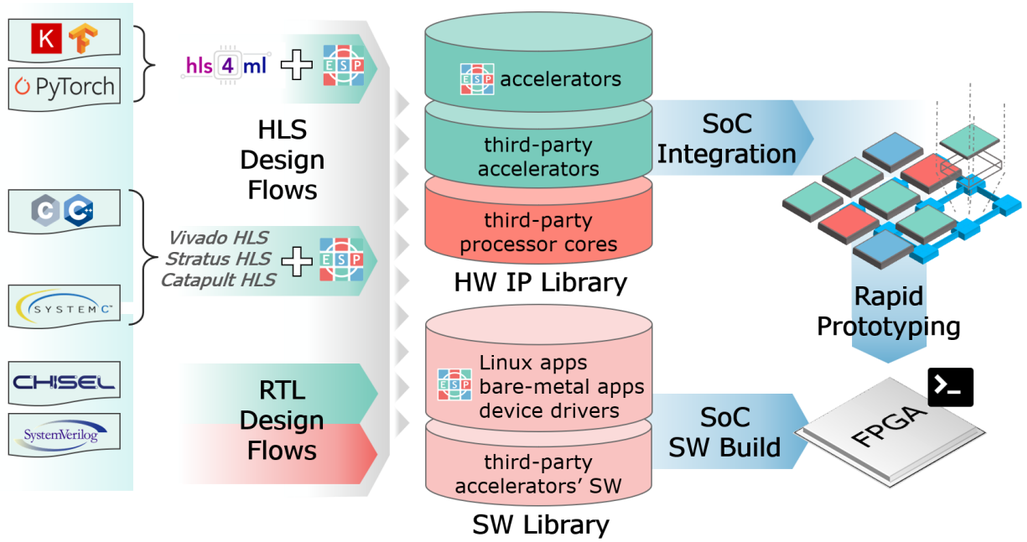
- New generation of “dataflow-like” approaches
  - Dynamatic: operation-level dataflow synthesis
  - HIDA: Hierarchical Dataflow approach, with some level of node optimizations
  - Stream-HLS: Enables graph-pipelining
  - FINN Compiler: also supports graph pipelining, with heavily quantized models

Research Opportunities enabled by an Open-Source Ecosystem

- SODA demonstrates how several Open-Source tools can seamlessly integrate
- SODA also provides initial support to commercial backends:
  - SODA-OPT generated LLVM IR can already be fed to Xilinx Vitis HLS
  - SODA also targets commercial ASIC logic synthesis tools
- Integration of proprietary tools, however, still is a significant challenges
- Significant opportunities in supporting:
  - Open-source intellectual property (IP) blocks as components in the resource libraries
  - Open-source system prototyping platforms
  - Open-source domain-specific FPGA generators to enable specialization starting from the high-level specifications (e.g., OpenFPGA)

Research Opportunities: System-Level Design

- Integrating with open-source fast prototyping platforms: Columbia University Embedded Scalable Platforms (ESP)
- SODA-OPT
  - MLIR is naturally modular and hierarchical
  - Can lower to multiple targets, including runtimes
- Bambu
  - Provides a fully open-source HLS backend for ESP
- Enables end-to-end fast prototyping from algorithmic concept to system implementation



Pacific Northwest NATIONAL LABORATORY

Research Opportunities: Physically-Aware HLS

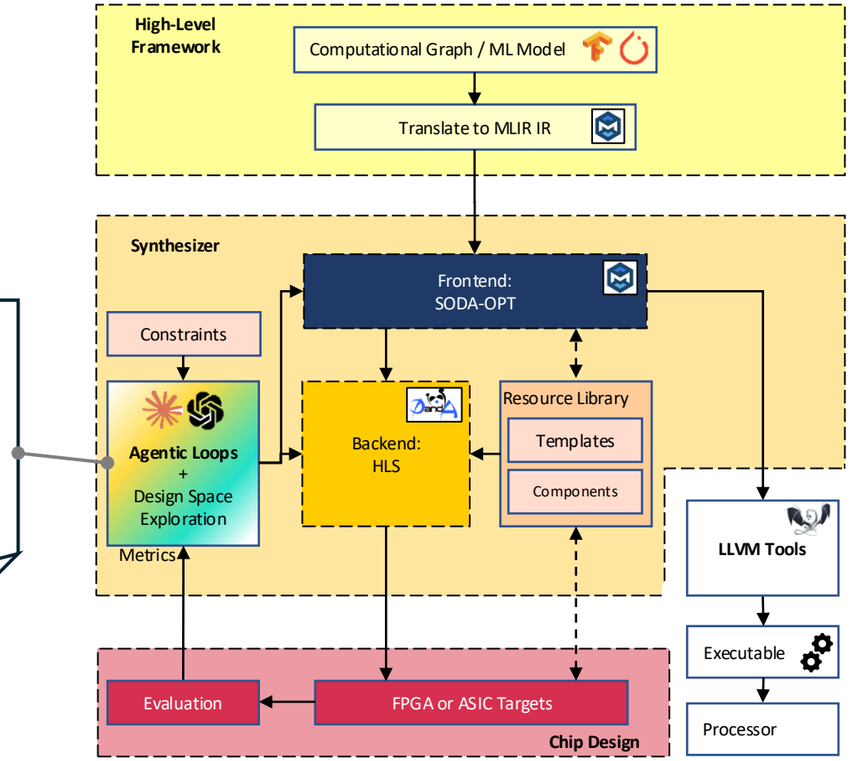
- An end-to-end flow creates opportunities to feed back physical information to higher abstraction levels
  - Schedule and bind operations with wiring delay awareness
  - Consider placement and routing when optimizations are applied
  - Integrate with memory compilers to move from integration of macros to customization of the memory subsystem
  - Leverage advanced interconnects for communication between modules (e.g., Networks-on-Chip)

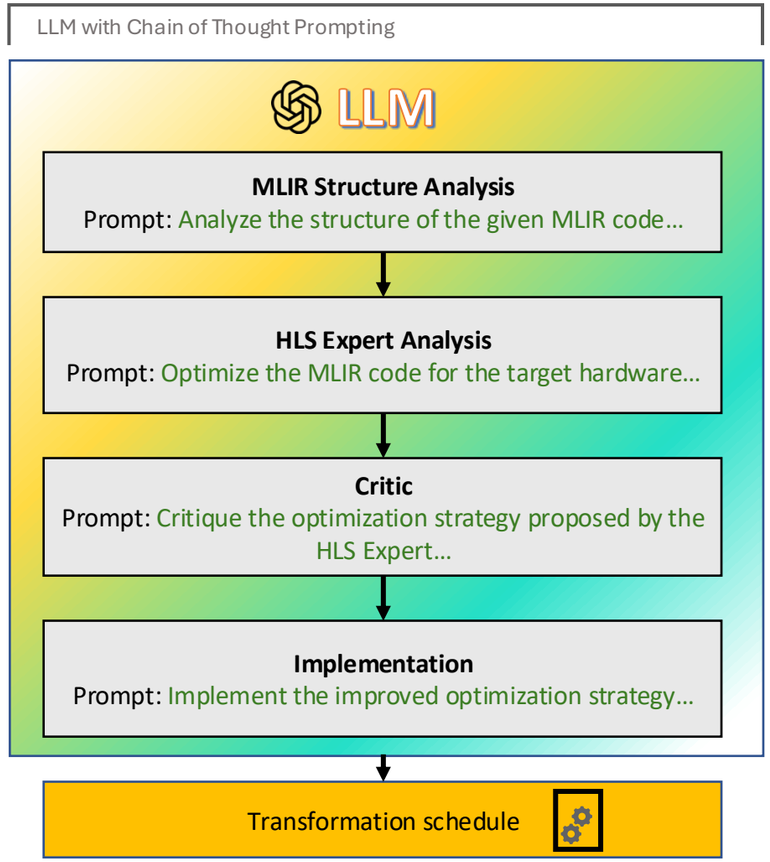
Research Opportunities: AI/ML with the Open-Source Hardware-Design Ecosystem

- SODA provides initial integration for several open-source tools
  - Example of open interfaces between SODA-OPT and Bambu and Bambu and OpenROAD with the Open-Source PDKs
- Need to further strengthen this integration, providing application programming interfaces (APIs) to access data at various stages of the synthesis
  - Provides the **necessary data** to train AI/ML models
  - Provides opportunity to predict effects of optimizations from one layer to the other
  - Provides opportunities for co-optimization, that do not necessarily only use neural networks
- Only through open interfaces we can strengthen the research methods
  - Proprietary tools could also be part of the flows, but they need to provide ways to exchange data and information

Agentic Loops Optimizing Software Defined Accelerators (ALO-SODA)

LLM Agents with access to compilers, synthesis and simulation results iterate over the design to meet Workload Requirements



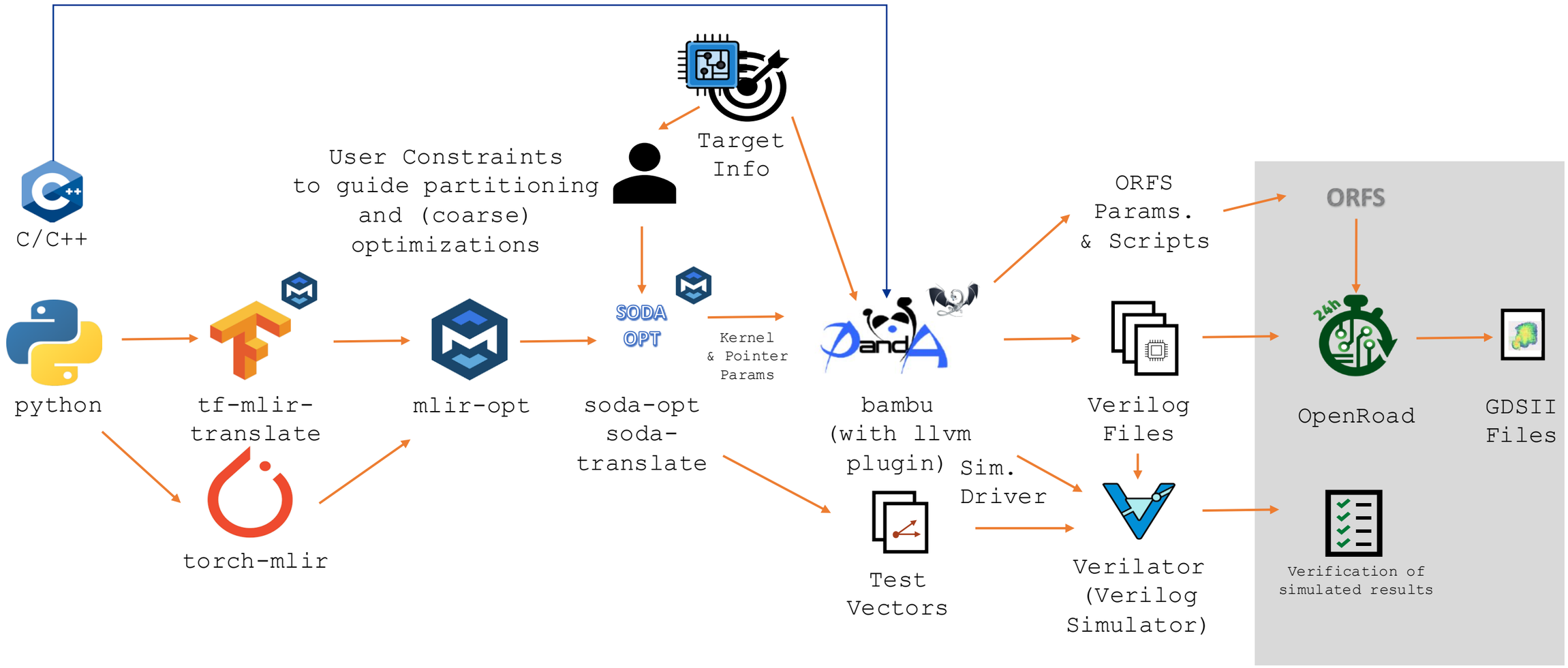


[M. Ramstad, N. Bohm Agostini, A. Tumeo, "LLM-Driven Optimization for High-Level Synthesis", ACM International Conference on Computing Frontiers, 2026, to appear]

ENCODE (End-to-End Co-design for Performance, Energy Efficiency and Security in AI-enabled Computational Science)

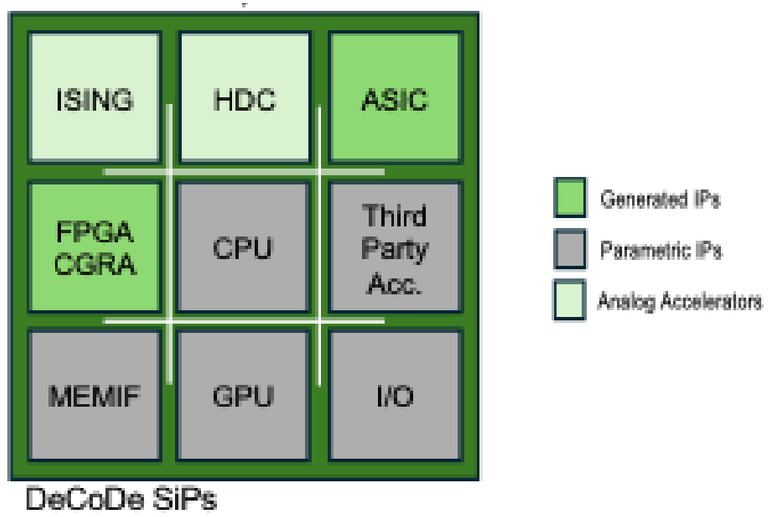
- https://www.pnnl.gov/projects/encode
- Develop an integrated end-to-end toolchain of **open-source hardware design tools** for custom **SiPs**
- Develop advanced **hardware synthesis methodologies and tools** to quickly convert high-level specifications and algorithmic formulations into specialized hardware (chiplets)
- Reduce design space exploration and design turnaround time by developing **scalable hardware co-optimization algorithms**
- Establish methodologies and testbeds for **pre- and post-fabrication testing and verification** of **chiplet-based designs**

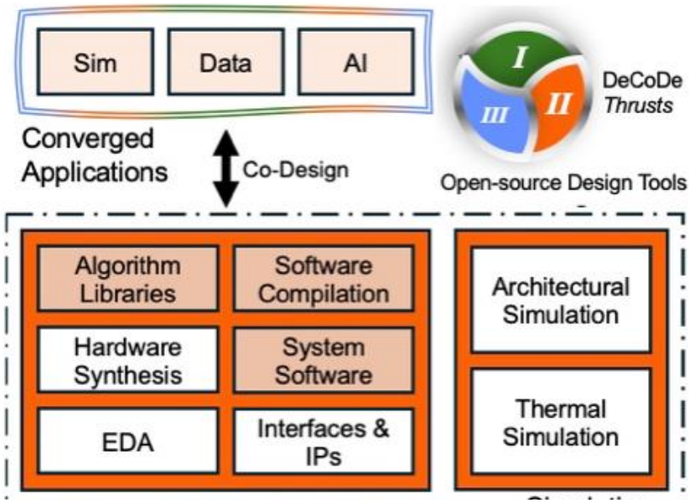
ENCODE End-to-End Pipeline



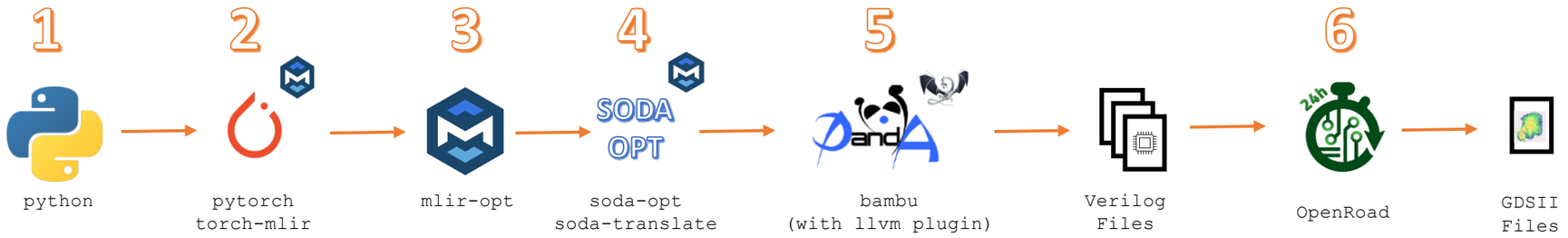
The Democratization of Co-Design (DeCoDe) for Energy-Efficient Heterogeneous Computing

- https://www.pnnl.gov/projects/decode
- Converged workloads also drive heterogeneous computing challenges: supercomputers-to-edge with integrated sensorsSystem-in-a-Package (SiP) advanced packaging where chiplets, are microprocessors, memory or network interfaces.
- Integration of analog accelerators, custom digital accelerators, and commodity CPU/GPUs into SiPs with software stack and algorithms is crucial to reach DOE energy-efficient performance goals.
- Balanced hardware-software co-design methodology, based on open standards, and open-source tools that can lower the cost of hardware R&D.
- With CHIPS Act investments in infrastructure for hardware prototyping and packaging, we have an opportunity to provide an open-source agile, low-cost hardware design and generation capability.





Hands-on Demo



1
- Write a simple DNN model in python

2
- Convert model to MLIR (`with torch_mlir`)
- Lower model to `TOSA` MLIR dialect

3
- Lower model to `linalg` MLIR dialect
- Visualize the intermediate representation

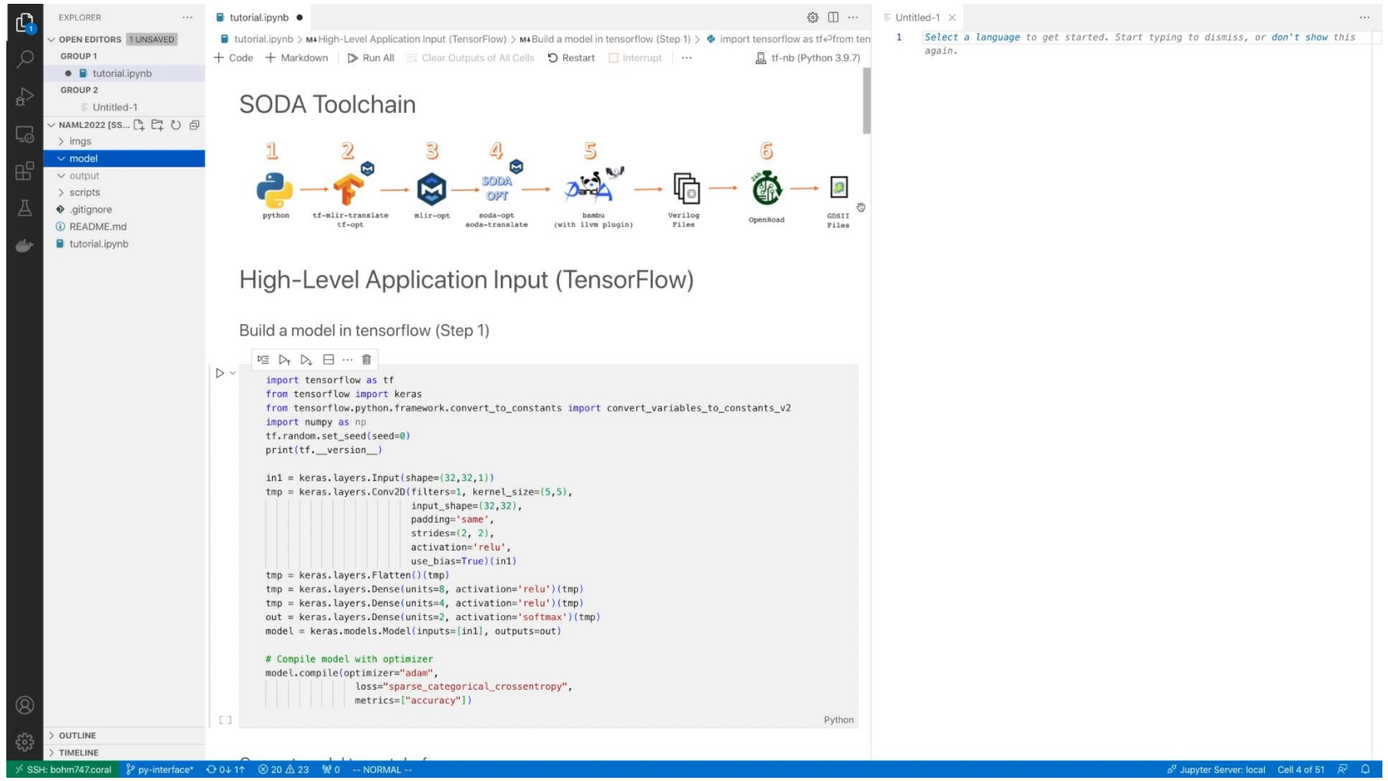
4
- Select MLIR code for custom accelerator generation
- Optimize kernel code and generate IR for Bambu

5
- Synthesize baseline and optimized code into Verilog
- Compare results

6
- Place and route synthesized code
- Visualize final GDSII files

https://github.com/pnnl/soda-benchmarks/tree/main/tutorials/pytorch

Hands-on Demo (Video)



https://youtu.be/RcqruxzPXp4

Pacific Northwest
NATIONAL LABORATORY

Thank you