DATE 26
DESIGN, AUTOMATION AND TEST IN EUROPE
THE EUROPEAN EVENT FOR ELECTRONIC SYSTEM DESIGN & TEST
20 – 22 APRIL 2026
VERONA, ITALY
PALAZZO DELLA GRAN GUARDIA

Open-Source Hardware Design: From High-Level Code to Silicon with Bambu and SODA

POLITECNICO MILANO 1863
DEPARTMENT OF ELECTRONICS INFORMATION AND BIOENGINEERING

Fabrizio Ferrandi
fabrizio.ferrandi@polimi.it

DandA

Session organization

ET03.1 Enabling Open-Source End-to-End Hardware Design Flows through High-Level Synthesis

- Fabrizio Ferrandi, Politecnico di Milano, Italy
- This session will present the tutorial scope and the tools that will be used in the hands-on exercises. It will then briefly introduce current and future research activities enabled by open-source High-Level Synthesis.

ET03.2 Synthesis and Optimization of Custom Accelerators with Bambu

- Serena Curzel, Politecnico di Milano, Italy
- This session is a hands-on experience showcasing basic and advanced Bambu features, enabled by Jupyter Notebooks hosted on Google Colab.

ET03.3 Agile hardware design with SODA: opportunities and challenges

- Antonino Tumeo, Pacific Northwest National Laboratory, United States
- This session introduces SODA-OPT, a frontend for system-level design in MLIR for HLS and beyond. It also features practical activities showcasing the SODA flow from high-level kernels to silicon.

Open-Source Hardware Design Flows for Custom Accelerators

- The open-source hardware ecosystem has matured, enabled by decentralized collaboration and rapid innovation.
- Open-source EDA tools, methodologies, and PDKs now support end-to-end design flows across multiple technology nodes.
- Growing needs in AI, data analytics, and emerging domains are driving demand for domain-specific accelerators with high performance and energy efficiency.
- High-Level Synthesis (HLS) is a key enabler to accelerate development, improve reproducibility, and reduce design cost and time.
- A **complete** open-source flow:
  - from high-level kernels (C/C++ or Python) to ASIC implementation using tools such as Bambu, SODA-OPT, and OpenROAD.
- Highlights ongoing initiatives/projects and concludes with current and future research directions in open-source EDA.

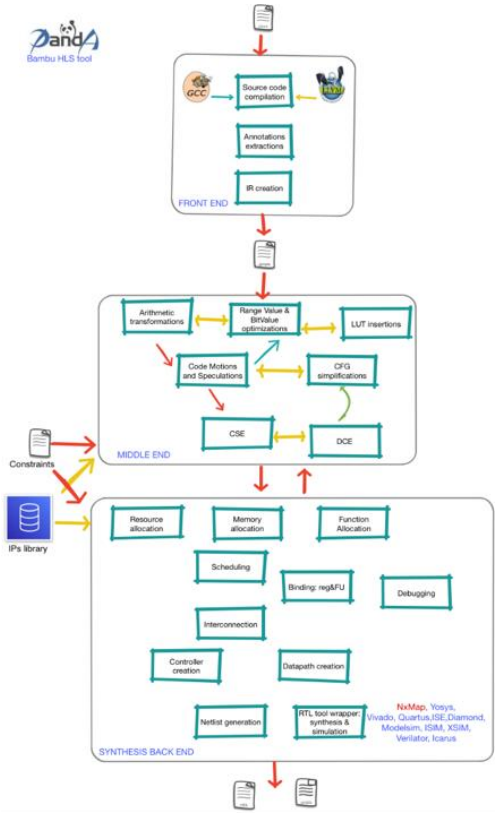
A bit of history about PandA

- PandA framework development started on 2004 as a support research infrastructure for PoliMi in the context of ICODES – FP6-IST EU-funded project
  - Parsing and analysis of TLM 2.0 SystemC descriptions (gcc v.3.5)
- In the hArtes EU-funded project (2006-2010), it was used to
  - Analyzing generic C-based application annotated with pragmas (OpenMP), Extracting parallel tasks, Estimating performance of embedded app
  - C-to-C rewriting
- Later, in Synaptic (2009-2013) and in Faster (2011-2014) EU-funded projects, logic- and high-level synthesis has been extended
  - Bambu (HLS tool) was first released in March 2012.
- HERMES – qualification of High pErformance pRogrammable Microprocessor and dEvelopment of Software ecosystem - 2021-2024
- EVEREST – dEsign enVironmEnt foR Extreme-Scale big data analytics on heterogeneous platforms - 2021-2024
- In these days, the project received further funding to extend its capability and support new targets:
  - ODE4EC - Open Design Environment for European Chips – European and Chips JU initiatives June 2026

Bambu: a modern HLS tool

- Bambu HLS
- Developed and maintained by the PandA group at Politecnico di Milano since 15+ years
- **Open-source**

- Synthesis flow:
- Compilation and optimization of the intermediate representations (IR)
- Allocation of resources
- Scheduling of operations
- Binding operations to resources
- Generation of synthesizable RTL

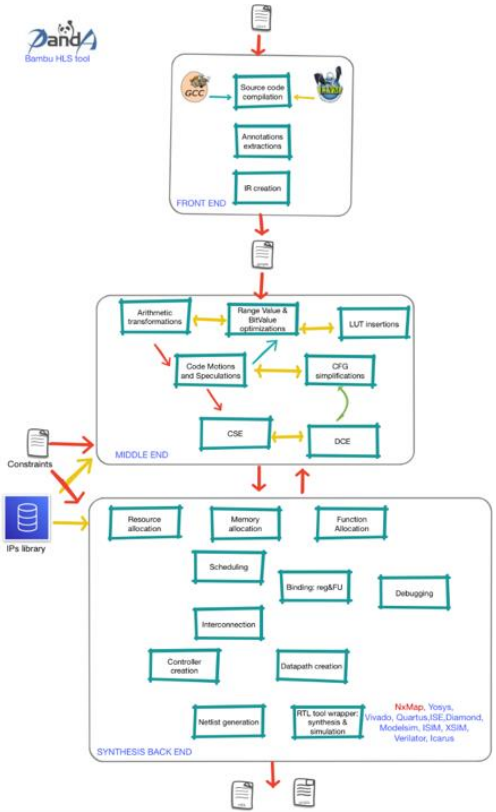


F. Ferrandi et al., Invited: Bambu: an Open-Source Research Framework for the High-Level Synthesis of Complex Applications, 58th ACM/IEEE Design Automation Conference (DAC), 2021.

Bambu: a modern HLS tool

Inputs:

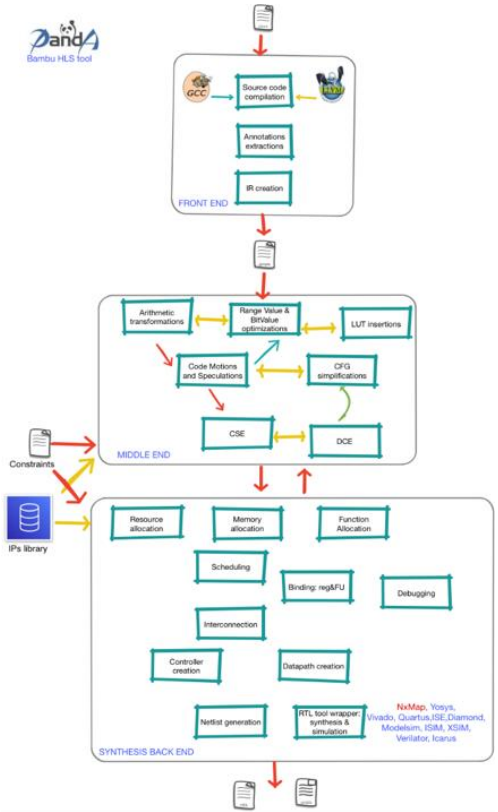
- C/C++/LLVM IR
  - Frontend compilation through GCC or Clang
  - Support for STL-like C++ structures
  - Support for fixed-point HLS types
- Command-line optimization directives and constraints, pragmas
- Library of functional units characterized for each target
  - Performance estimation essential for scheduling and optimization



Bambu: a modern HLS tool

Outputs:

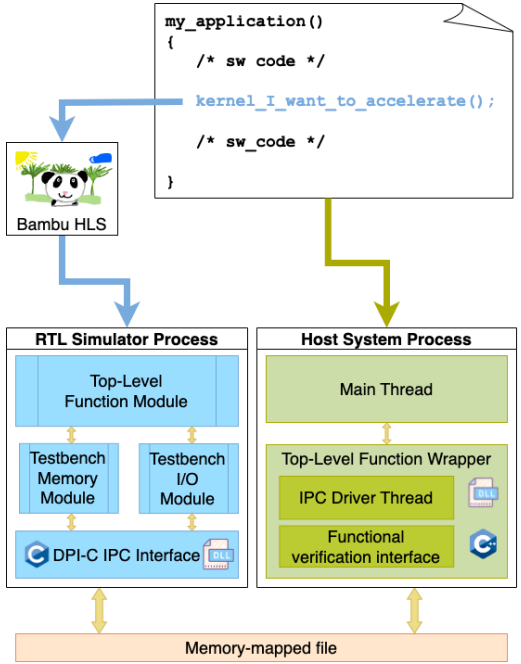
- Synthesizable Verilog/VHDL
  - Target-specific
  - FPGA targets from AMD/Xilinx, Intel/Altera, Lattice, NanoXplore
  - ASIC targets thorugh OpenROAD (Nangate45, ASAP7)
- Automatically generated testbench
  - RTL simulation with Verilator/Modelsim/XSIM
- Scripts for logic synthesis/implementation
  - Vivado/Quartus/Diamond…



Accelerator – host co-simulation

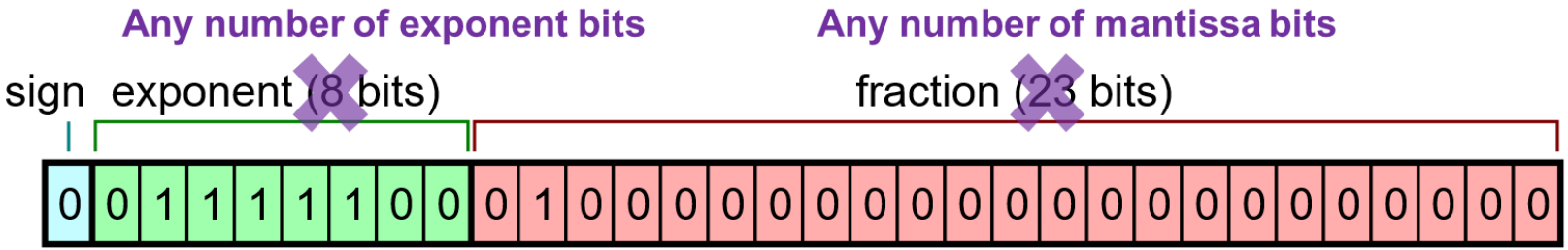
Outputs:

- HW/SW co-simulation framework
  - RTL testbench and infrastructure automatically generated by Bambu
  - Reports number of clock cycles with the given input data
  - Compares HW results with SW, or defers verification to the testbench code Target-specific



M. Fiorito et al., Augmented Co-Simulation for Fast Functional and System-Level Verification of HLS Accelerators, 44th IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2025.

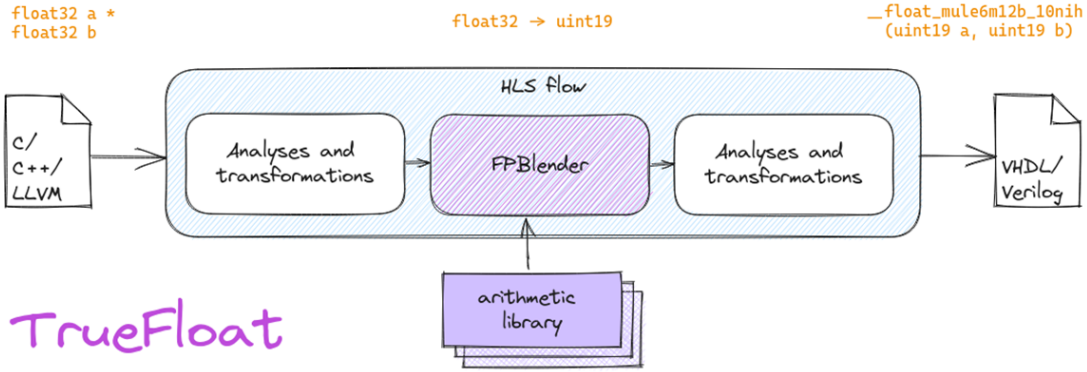
Custom floating-point formats



But also:

- Custom exponent bias
- Rounding to nearest even or truncation
- IEEE exceptions, saturation, or overflow
- Hidden one, or not
- Subnormals, or not
- Dynamic or static sign

TrueFloat



- Users specify the required format, Bambu generates optimized arithmetic units
- Integrated within the HLS flow -> QoR higher than inserting «black box» IPs
- Effortless translation between encodings through command-line options

M. Fiorito et al., TrueFloat: A Templatized Arithmetic Library for HLS Floating-Point Operators. In Embedded Computer Systems: Architectures, Modeling, and Simulation. SAMOS 2023.

Interface generation

Supported protocols:

- Minimal Memory Interface (simple Bambu bus)
- FIFO, AxiStream
- Handshake
- Array
- AXI4-Master

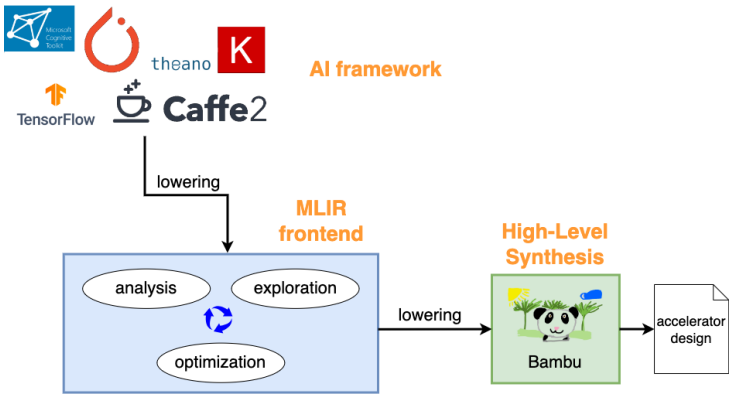
(Mostly) compatible with Vitis HLS interfaces

Requested by the user through `--generate-interface=INFER` and either

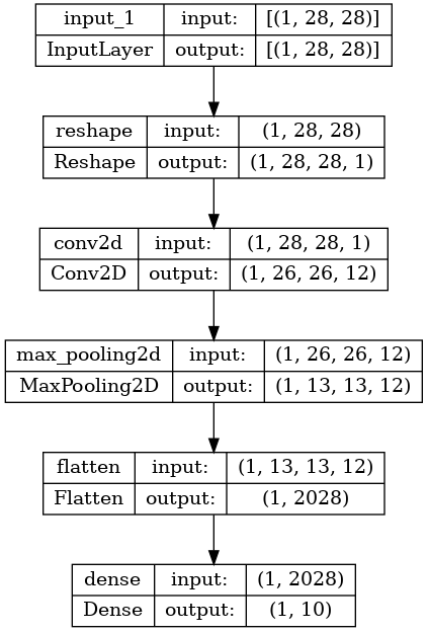
```
#pragma HLS bus mode = m_axi
#pragma HLS interface port = edges mode = bus
#pragma HLS interface port = level mode = bus
void bfs(edge_t* edges, level_t* level, ...)
```

Compiler-driven innovation: MLIR

- Multi-Level Intermediate Representation (MLIR), infrastructure for domain-specific compilers
  - Used predominantly for AI
- MLIR -> LLVM dialect -> LLVM IR -> Bambu
  - Any design written in MLIR can be synthesized!



CNN acceleration on FPGA

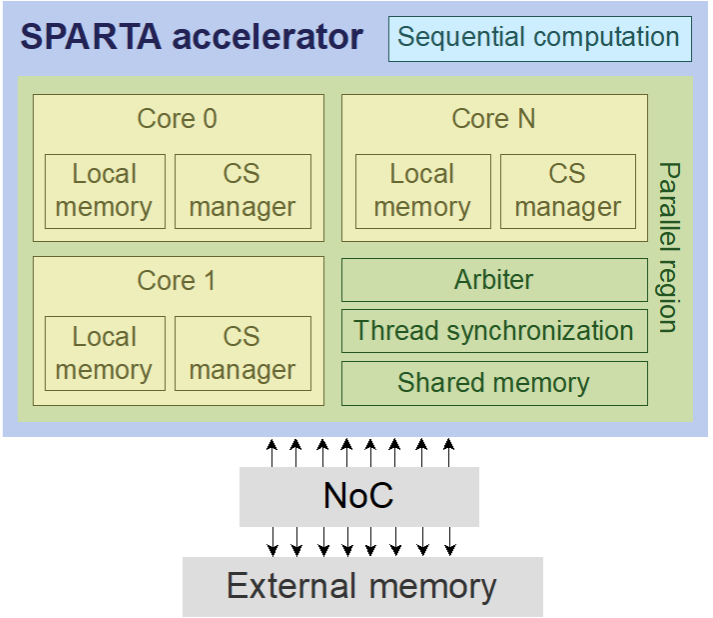


	NG-Ultra
LUTS	4627
Registers	5714
Frequency	45.7 MHz
DSP	54
MEM	34
cycles	169,649
latency	3.7ms

- Neural network trained and quantized to 8-bit to reduce area and improve hardware performance
- TFLite to MLIR export
- Bambu synthesis

Synthesis of PARallel multi-Threaded Accelerators (SPARTA)

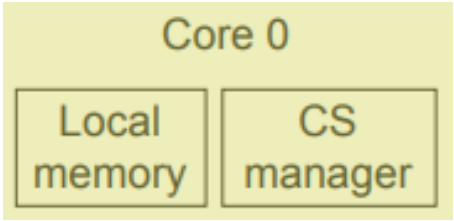
- Starts from high-level languages (C/C++) annotated with OpenMP pragmas
  - Parallel, for, sections
  - Reduction, shared, private, firstprivate
  - Critical, barrier
- Produces an accelerator with multiple cores that can execute concurrently



G. Gozzi et al., SPARTA: High-Level Synthesis of Parallel Multi-Threaded Accelerators. In: ACM Trans. Reconfigurable Technol. Syst. 18.1 (Dec. 2024)

SPARTA synthesis flow

```
void dot_product(int A[M], int B[M], int res[M]){
   int sum = 0;
#pragma omp parallel num_threads(M) for
   for(i = 0; i < M; i++){
      prod(A[i], B[i], res[i]);
   }
}
```

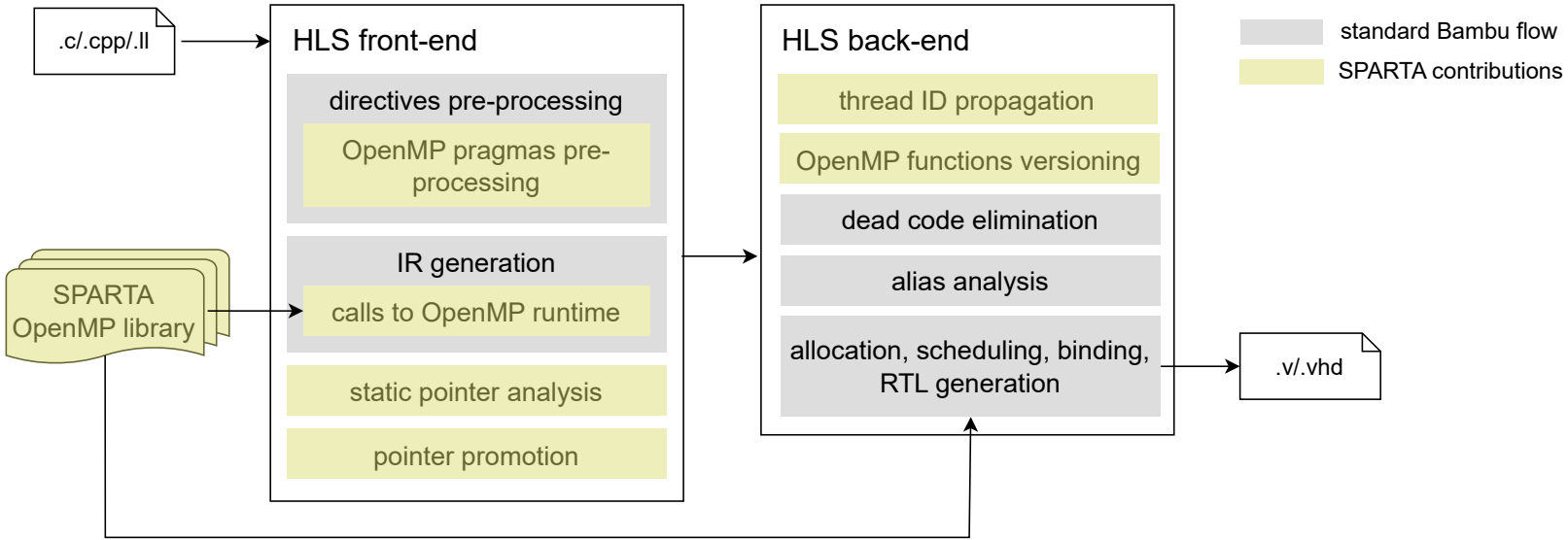



```
define i32 @dot_product (...) {
   ...
   call void __kmpc_push_num_threads(M)
   call void __kmpc_fork_call(.omp_outlined., ...)
   ...
}
define void @ .omp_outlined.(...) {
   %6 = call i32 __kmp_bambu_tid_from_gtid(i32 %0)
   ...
   call void prod(...)
   call void __kmp_bambu_barrier_reached(i32 %6)
   call void __kmp_bambu_wait_all_threads()
   ...
}
```

- Each outlined function is mapped to a core, which can be shared through *context switching*
  - Hide external memory access latency

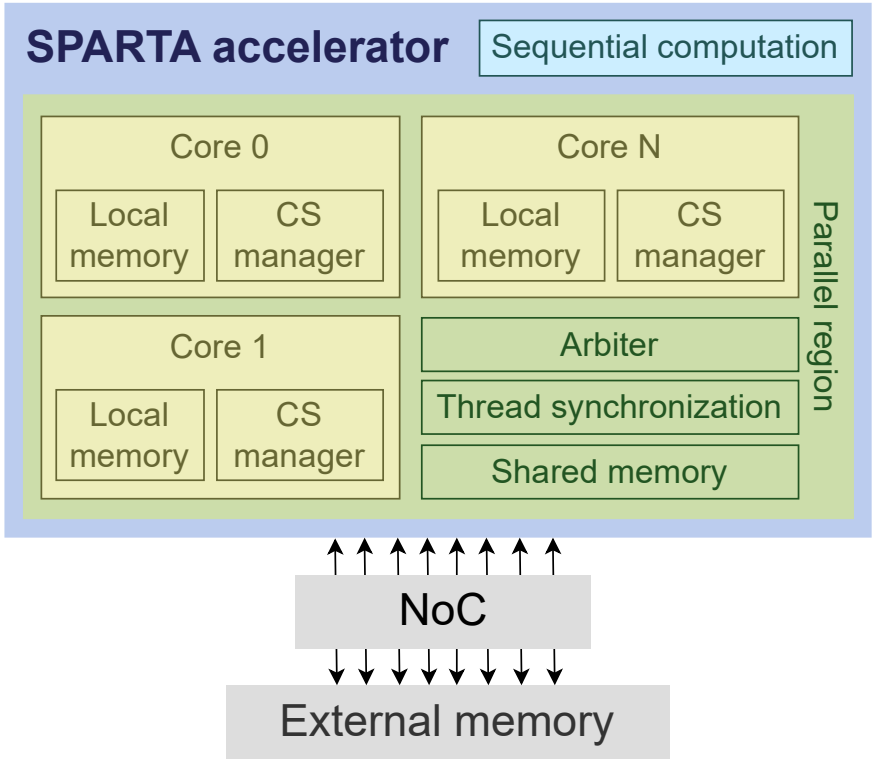
SPARTA synthesis flow

- **Integration to the standard synthesis flow of Bambu exploiting the LLVM OpenMP runtime**



SPARTA architecture

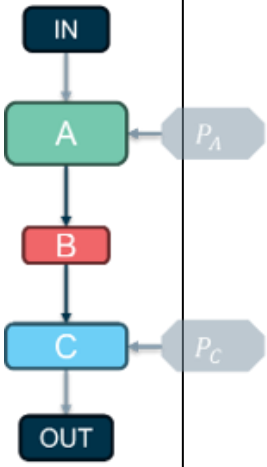
- Multiple parallel cores
- Hardware for arbitration and synchronization
- Efficient implementation of reductions
- Configurable number of channels towards external memory and custom NoC
- Works very well for irregular applications

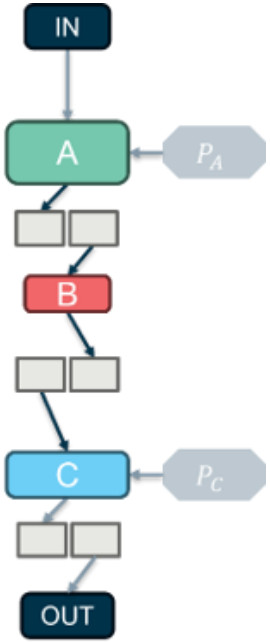


Dataflow

```
class SimpleSystem {
   AddBlock<5> A;
   MulBlock B;
   SubSystem C;
   ac_channel<int> x, y;

 public:
   void top(ac_channel<int>& in1, ac_channel<int>& in2,
            ac_channel<int>& in3,
    ac_channel<int>& out) {
      #pragma HLS dataflow
      A.compute(in1, x);
      B.compute(x, in2, y);
      C.compute(y, in3, out);
   }
};
void dataflow_top(ac_channel<int>& in1, ac_channel<int>& in2,
                  ac_channel<int>& in3,
        ac_channel<int>& out) {
   static SimpleSystem sys;
   sys.top(in1, in2, in3, out);
}
```





Dataflow

- Dataflow support
- `ac_channel` & `hls::stream`
- FIFO depth could be controlled by pragmas
- Struct/array passed as template parameter supported
- Data field member support could be improved
- `ac_int` and `ac_fixed` supported
  - Synthesis efficiency improved when PandA-Bambu is used with Clang16

OpenRoad integration

- OpenROAD is a production-grade open-source RTL-to-GDSII flow for automated digital chip physical design (place & route), enabling fast and reproducible ASIC implementation.
- Bambu integration of OpenRoad as RTL synthesis backend
  - Docker-based
  - Initial support of SRAM synthesis
  - Initial support of pipelined components (2Ghz)

Ginevra project

- hls4ml translates ML models into HLS-based FPGA firmware for ultra-low-latency inference, lowering the barrier to deploying neural networks in hardware.
- Bambu integration as an HLS4ML backend
  - Support low-latency design
  - LUT-based compression
  - Improved dataflow support