

28–29 Nov 2023

European Space Research and Technology Centre (ESTEC)



# **Exploring the Latest High-Level Synthesis Results with Bambu: Insights from the HERMES Project's Use Cases**

**Fabrizio Ferrandi, Claudio Barone, Serena Curzel, Michele Fiorito, Giovanni Gozzi**

**Politecnico di Milano, Italy,**



# Outline

- **Introduction**
- **Bambu HLS**
- **Bambu extensions**
- **Conclusion**

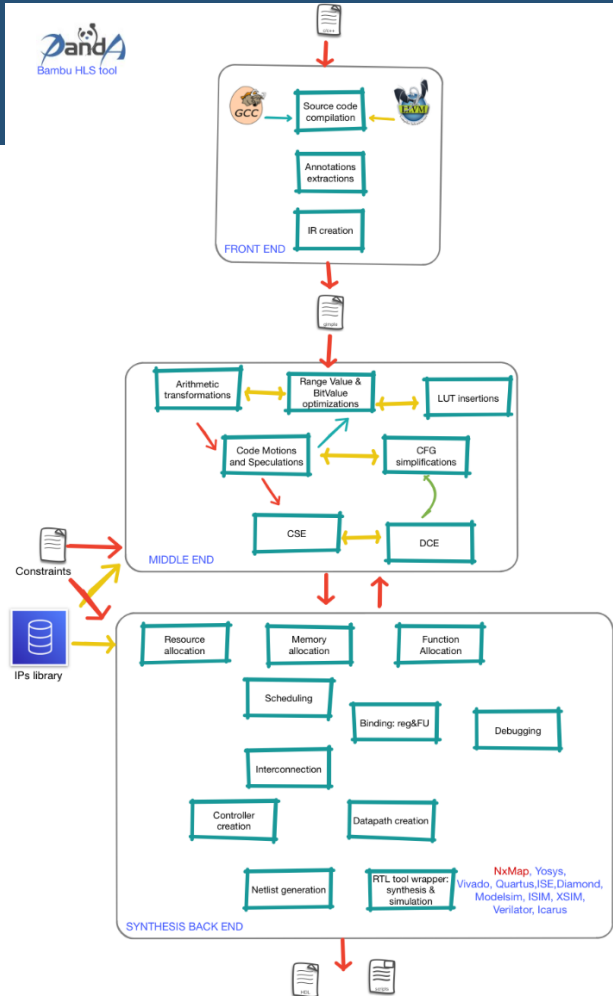
# HERMES project

- **H2020-funded project started in March 2021**
- **It aims at validating and evaluating a state-of-the-art rad-hard FPGA according to the standards of the European Space Components Coordination (ESCC), and at integrating design and manufacturing technologies needed to deliver high-reliability applications running on radiation-hardened integrated circuits**
- **Consortium is composed by**
  - *NanoXplore, France,*
  - *Politecnico di Milano, Italy,*
  - *Fent Innovative Software Solutions – FentISS, Spain,*
  - *Thales Alenia Space SAS, France,*
  - *STMicroelectronics Grenoble SAS, France,*
  - *Airbus Defence And Space SAS, France*

# Bambu HLS

- HLS tools simplify the implementation of accelerators on FPGA
- HLS starts from high-level languages (C/C++)
  - Optimizes the intermediate representations
  - Allocates resources
  - Schedules operations
  - Binds them to the resources
  - And generates RTL descriptions for synthesis tools

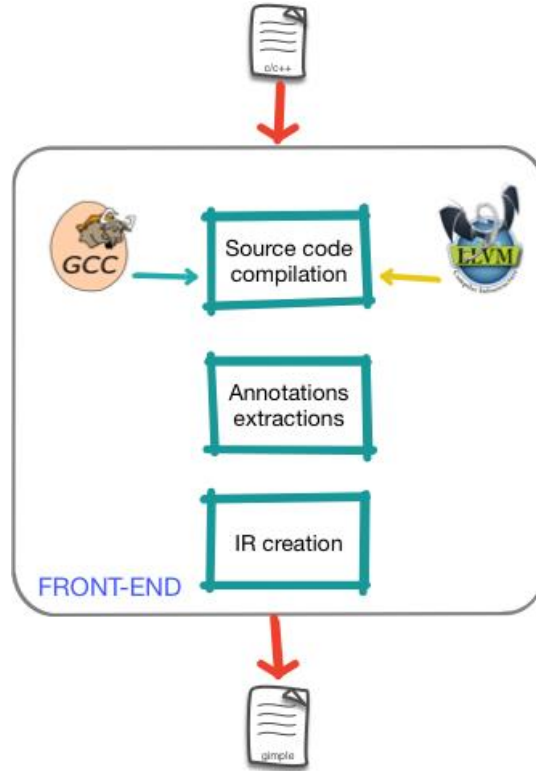
The increased performance offered by FPGAs is made available also to software developers that do not have hardware design expertise



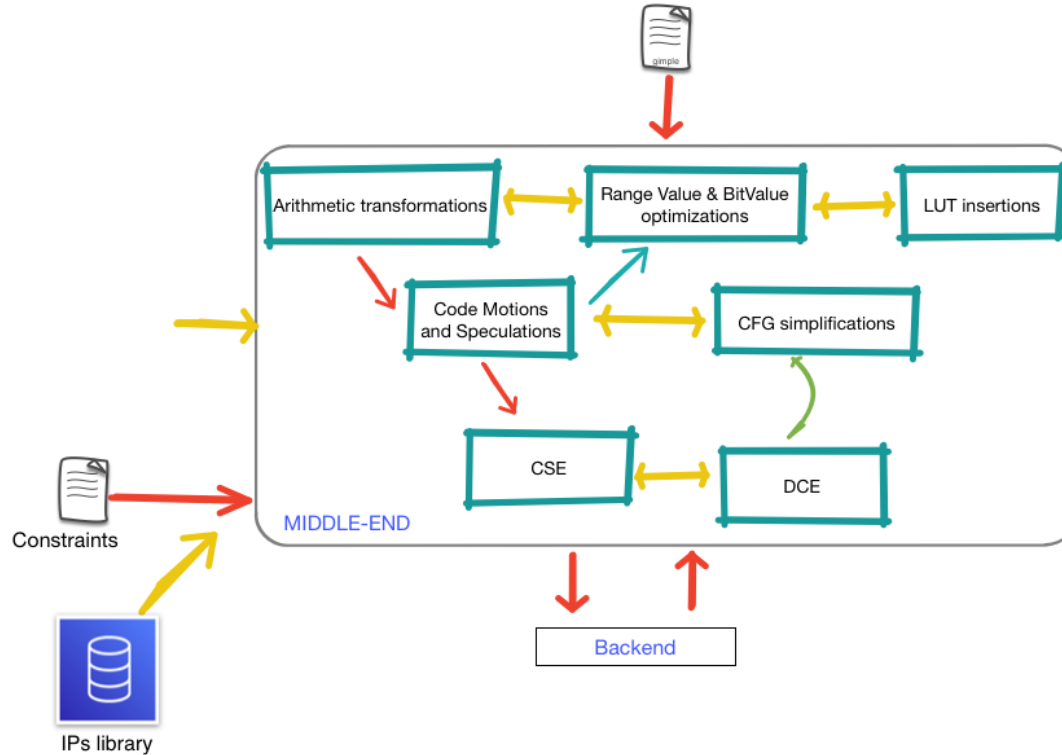
# Bambu: an example of modern HLS tools

- **Open-source HLS tool developed at Politecnico di Milano (Italy)**
  - **Front-end Input: interfacing with GCC/CLANG-LLVM for parsing C code**
    - **Complete support for ANSI C (except for recursion)**
      - **Support for pointers, user-defined data types, built-in C functions, etc..**
    - **Source code optimizations**
      - **may alias analysis, dead-code elimination, hoisting, loop optimizations, etc...**
  - **Target-aware synthesis**
    - **Characterization of the technology library based on target device**
  - **Verification**
    - **Integrated testbench generation and simulation**
      - **automated interaction with Verilator, Xilinx Xsim, Mentor Modelsim**
  - **Back-end: Automated interaction with commercial synthesis tools**
    - **FPGA: Xilinx ISE, Xilinx Vivado, Altera Quartus, Lattice Diamond, NanoXplore**
    - **ASIC: OpenRoad (Nangate 45, ASAP7)**

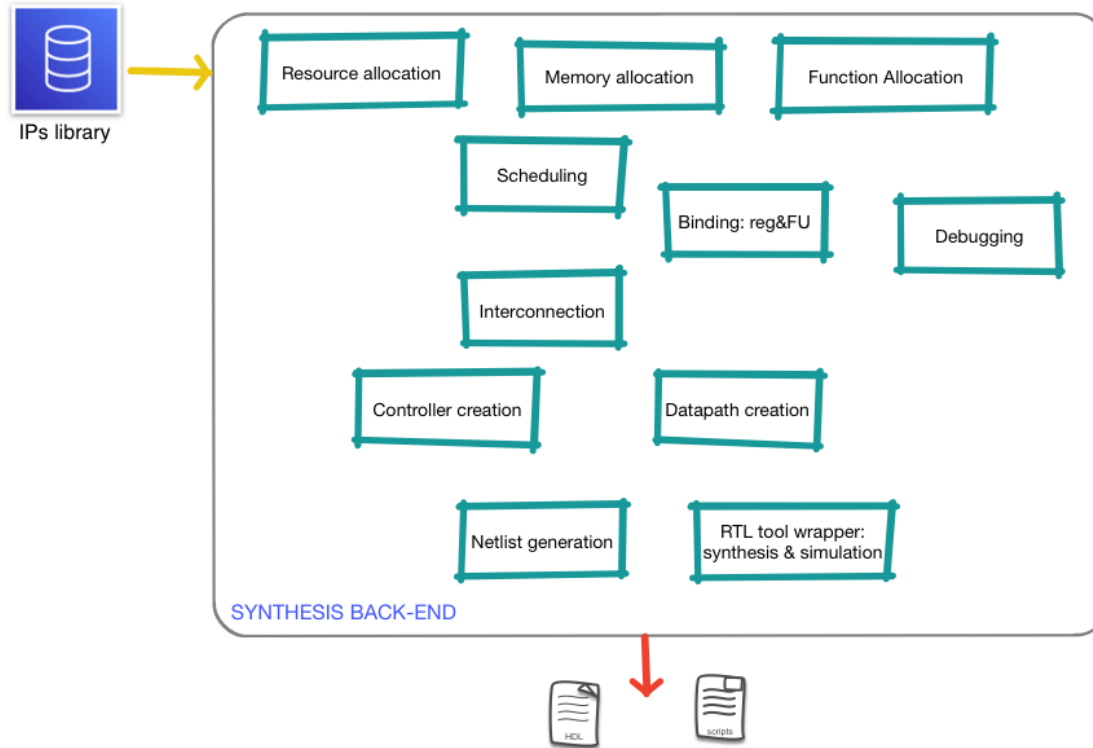
# Bambu: front-end



# Bambu: middle-end



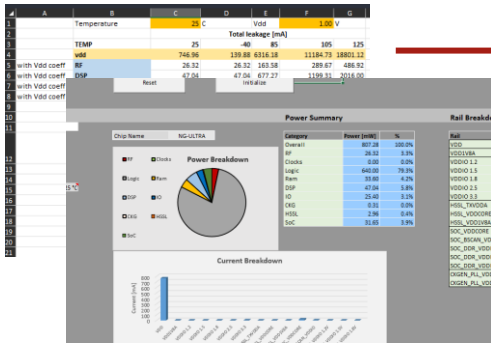
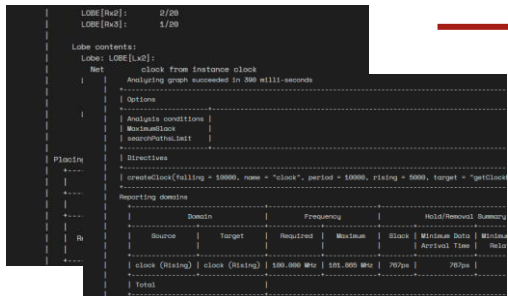
# Bambu: back-end





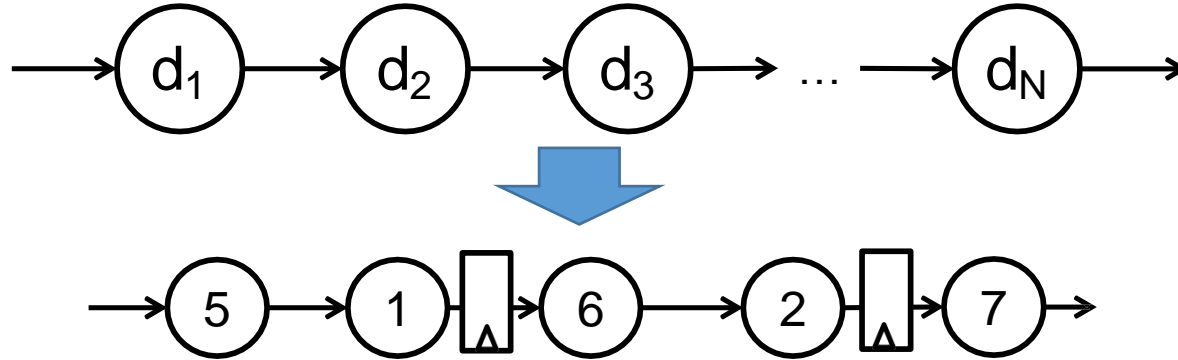
# Bambu – NX integration

- **Seamless integration between Bambu and Impulse from NX**
  - automatic generation of backend synthesis scripts
- **Bambu IP resource library characterized with respect to rad-hard NX FPGAs**
  - Resource occupation and latency under different design constraints
  - Automatically performed with PandA Eucalyptus tool



```
<?xml version="1.0"?>
<document>
  <application>
    <section stringID="NANOXPLORE_SYNTHESIS_SUMMARY">
      <item stringID="NANOXPLORE_FE" value="696"/>
      <item stringID="NANOXPLORE_LUTS" value="360"/>
      <item stringID="NANOXPLORE_REGISTERS"
value="404"/>
      <item stringID="NANOXPLORE_MEM" value="2"/>
      <item stringID="NANOXPLORE_IOPIN" value="292"/>
      <item stringID="NANOXPLORE_DSPS" value="0"/>
      <item stringID="NANOXPLORE_POWER" value="0.833"/>
      <item stringID="NANOXPLORE_SLACK" value="2.929"/>
    </section>
  </application>
</document>
```

# Bambu – NX integration



- **Performance estimation essential for**
  - Aggressive scheduling
  - Pipelining
  - Code transformations
- **Bambu IP library customized with respect to**
  - DSPs
  - NG-ULTRA fabric True Dual Port RAMs

# Bambu – NX integration

- Added support for NanoXplore radiation-hardened FPGAs with device-specific RTL component library characterization for:
  - NG-MEDIUM (nx1h35S)
  - NG-LARGE (nx1h140tsp)
  - NG-ULTRA (nx2h540tsc)

```
<cell>
  <name>fp_plus_expr_FU_0_32_32_100</name>
  <attribute name="area" value_type="float64">649</attribute>
  <attribute name="REGISTERS" value_type="float64">194</attribute>
  <attribute name="SLICE_LUTS" value_type="float64">457</attribute>
  <template name="fp_plus_expr_FU" parameter="0 32 32 100"/>
```

```
<cell>
  <name>fp_plus_expr_FU_0_32_32_100</name>
  <attribute name="area" value_type="float64">649</attribute>
  <attribute name="REGISTERS" value_type="float64">194</attribute>
  <attribute name="SLICE_LUTS" value_type="float64">457</attribute>
```

```
<cell>
  <name>fp_plus_expr_FU_0_32_32_100</name>
  <attribute name="area" value_type="float64">638</attribute>
  <attribute name="REGISTERS" value_type="float64">150</attribute>
  <attribute name="SLICE_LUTS" value_type="float64">464</attribute>
  <template name="fp_plus_expr_FU" parameter="0 32 32 100"/>
  <characterization_timestamp>2022-05-19T10:41:00</characterization_timestamp>
  <operation operation_name="plus_expr" commutative="1"
    supported_types="REAL:32" pipe_parameters="100" cycles="2"
    initiation_time="1" stage_period="19.088999999999999"/>
</cell>
```

# Use Cases

- **HERMES** project is validating the Bambu HLS tool through typical space use cases
  - image and vision processing algorithms,
  - software-defined algorithms,
  - artificial intelligence applications

# Bambu – extended features

- **Added support to AXI4 master and AXIS interfaces**
  - **Easier integration with ARM processor on the NG-ULTRA board**
  - **Easy way to integrate existing IPs**
- **Automatic generation of AXI testbench supported**
- **Memory latency can be configured**
- **Unaligned accesses are supported**
- **AXI4 burst transactions supported**

# Bambu – extended features

	AXI4 Memory Copy	AXI4 Memory Read
LUTS	1072	276
Registers	910	142
Frequency	114 MHz	189.43 MHz

```
#pragma HLS_interface a m_axi direct
#pragma HLS_interface b m_axi direct
int __attribute__((noinline)) vector_copy_plus(int* a, int* b, int elem_number)
{
    for(int i = 0; i < elem_number; i++)
    {
        b[i] = a[i] + 1;
    }
    return b[0];
}
```

```
#pragma HLS_interface data m_axi direct
int read(int * data)
{
    return *data;
}
```

# Bambu – extended features

- **Support for caches on AXI interfaces**
  - Customizable cache and line size
  - Support for different write policies
  - Support for associative caches
  - Support for different replacement policies
  - Support for larger AXI xDATA signal size
  - Support for pipelined write transactions
  - Automatic cache flush at the end of the computation
  - Includes simulation-only hit/miss counters

# Bambu – extended features

- **C++ FIFO interface support:**

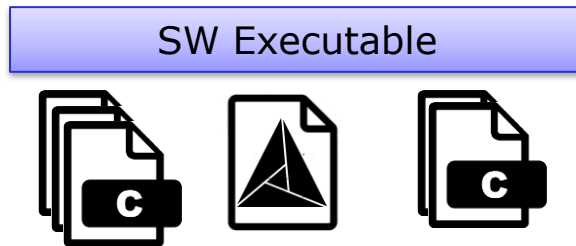
- `ac_channel<T>`
- `hls::stream<T>`

```
void sum3numbers(ac_channel<ap_uint<64>>& a,  
                ac_channel<ap_uint<64>>& b,  
                ac_channel<ap_uint<64>>& c,  
                ac_channel<ap_uint<64>>& d)  
{  
    int i;  
    for(i = 0; i < 8; ++i)  
        d.write(a.read() + b.read() + c.read());  
}
```



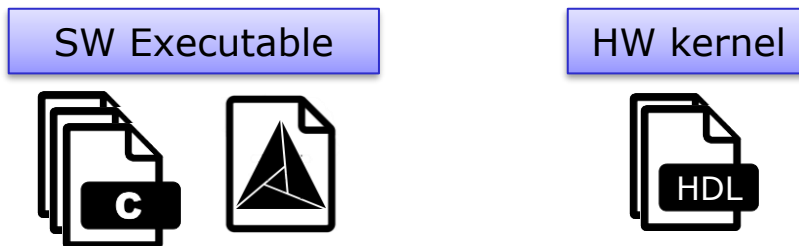
# Standard verification flow

- **Starting point**

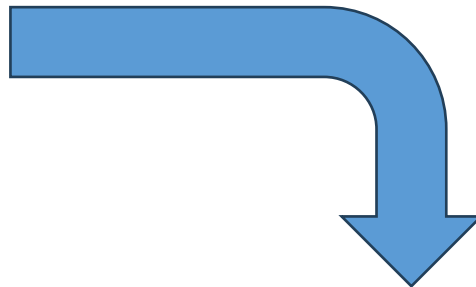
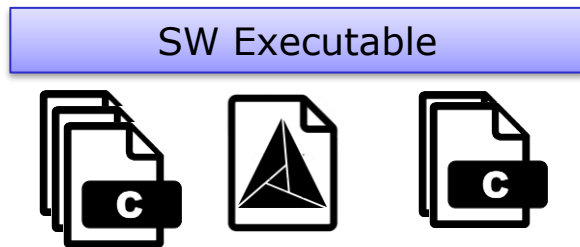


- **Expected result**

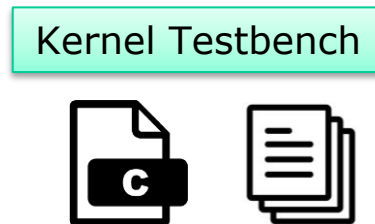
- **Verified software application with verified hardware-accelerated kernel**



# Standard Verification flow

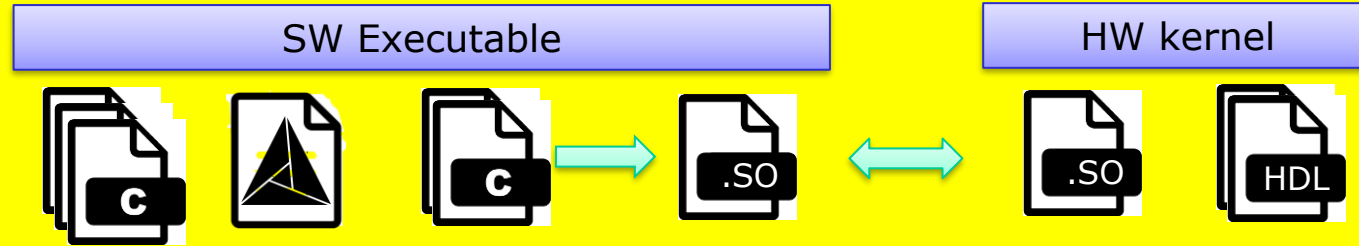


- Testbench built as unit testing module
- Kernel I/O test file generation
- Error-prone
- Separated from use case application



# Bambu verification approach

1. Standard verification flow is now supported even by Bambu
2. **Bambu additional verification approach**

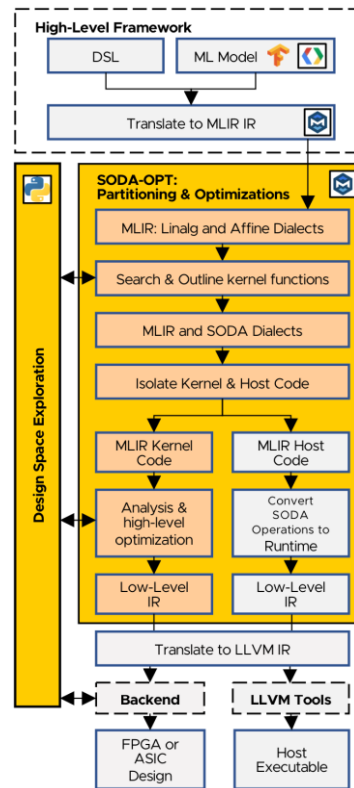


- **Application instrumented to compare software and HW execution using the original application**
- **DPI-C interface used to connect the SW and the HW worlds**

# Integrating ML-design flow in Bambu

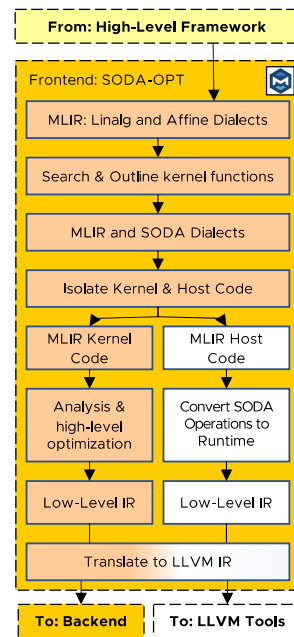
- Research ongoing on DSLs to synthesize accelerators for machine learning-based inference.
- We did some experiments with SODA-OPT framework
  - Aiming at optimizing one of the TAS use-cases.
- SODA-OPT jointly developed by PNNL, Northwestern University and Politecnico di Milano

Serena Curzel, Nicolas Bohm Agostini, Vito Giovanni Castellana, Marco Minutoli, Ankur Limaye, Joseph B. Manzano, Jeff Zhang, David Brooks, Gu-Yeon Wei, Fabrizio Ferrandi, Antonino Tumeo: End-to-End Synthesis of Dynamically Controlled Machine Learning Accelerators. IEEE Trans. Computers 71(12): 3074-3087 (2022)



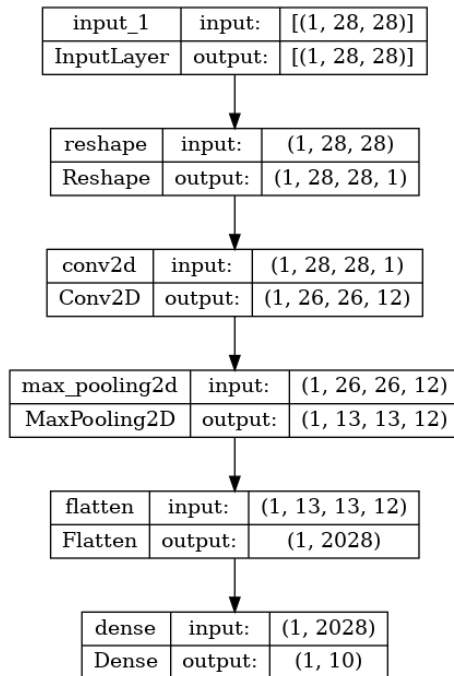
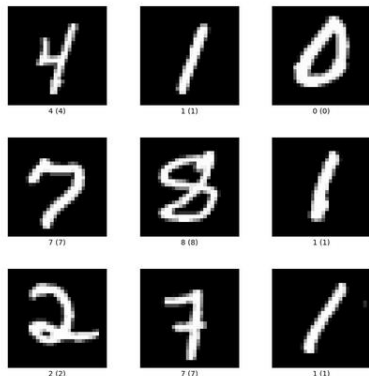
# SODA-OPT: Frontend and High-Level IR

- **SODA-OPT: Search, Outline, Dispatch, Accelerate** frontend optimizer “generates” the SODA High-Level IR
- **Employs and embraces the MLIR framework**
  - MLIR: Multi-Level Intermediate Representation
  - Used in TensorFlow, TFRT, ONNX-MLIR, NPYComp, 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



<https://gitlab.pnnl.gov/sodalite/soda-opt>

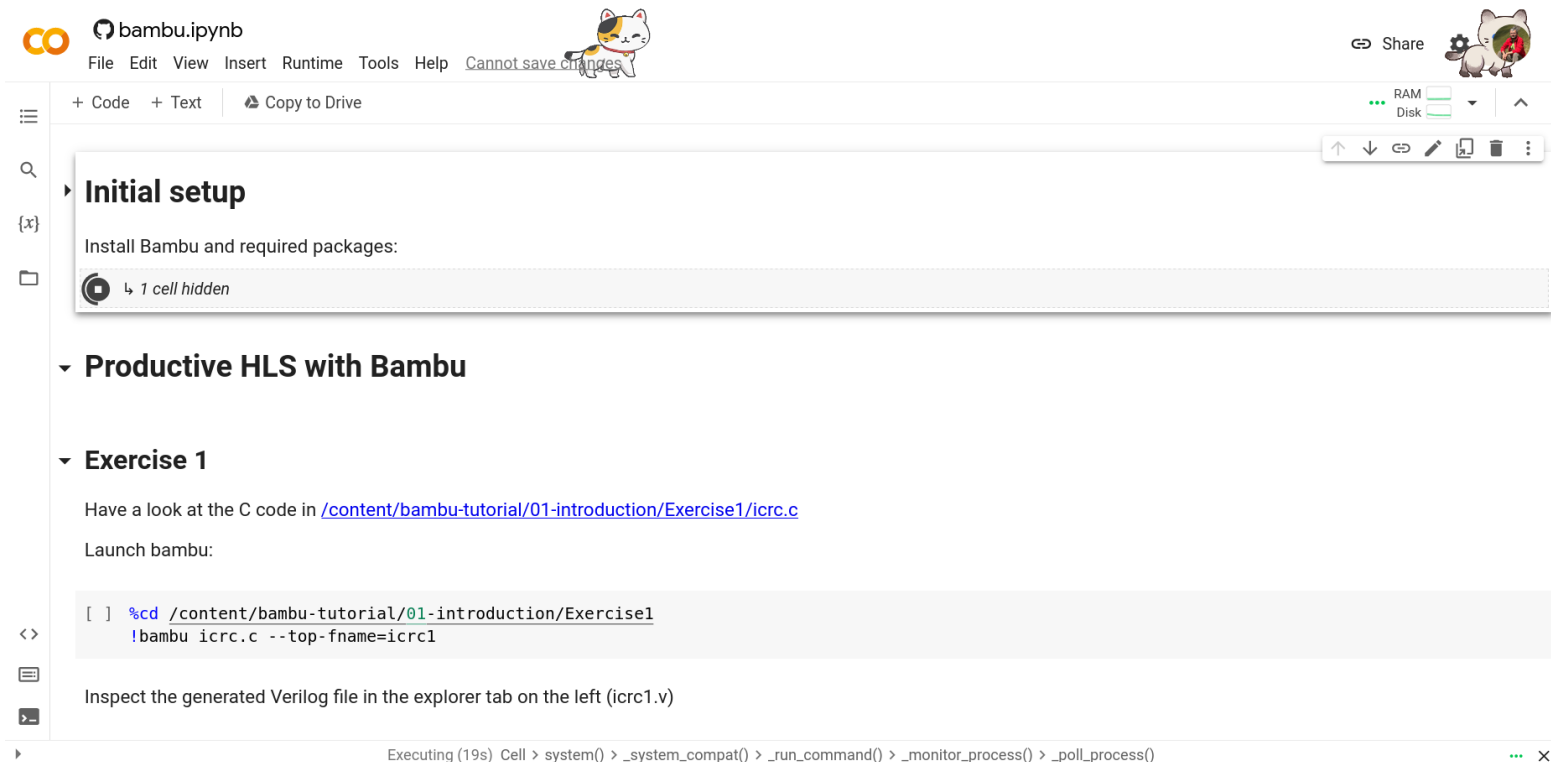
# MNIST example



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

- **8-bit quantized**
- **Tensorflow 2.15**
- **Clang 18**
- **MLIR based design flow**

# Google Colab notebook



The screenshot shows a Google Colab notebook interface. At the top, the title bar reads 'bambu.ipynb' with a 'Cannot save changes' warning. The menu bar includes File, Edit, View, Insert, Runtime, Tools, and Help. On the right, there is a 'Share' button and a cat icon. Below the menu bar, the toolbar shows '+ Code', '+ Text', and 'Copy to Drive'. The left sidebar contains icons for file explorer, search, and a list of cells. The main content area is divided into sections: 'Initial setup' with the instruction 'Install Bambu and required packages:' and a note '1 cell hidden'; 'Productive HLS with Bambu'; and 'Exercise 1' which includes the text 'Have a look at the C code in [/content/bambu-tutorial/01-introduction/Exercise1/icrc.c](#)' and 'Launch bambu:'. A code cell is shown with the command: 

```
[ ] %cd /content/bambu-tutorial/01-introduction/Exercise1
!bambu icrc.c --top-fname=icrc1
```

 Below the code cell, it says 'Inspect the generated Verilog file in the explorer tab on the left (icrc1.v)'. At the bottom, the status bar indicates 'Executing (19s) Cell > system() > \_system\_compat() > \_run\_command() > \_monitor\_process() > \_poll\_process()'.

bambu.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

**Initial setup**

Install Bambu and required packages:

1 cell hidden

**Productive HLS with Bambu**

**Exercise 1**

Have a look at the C code in [/content/bambu-tutorial/01-introduction/Exercise1/icrc.c](#)

Launch bambu:

```
[ ] %cd /content/bambu-tutorial/01-introduction/Exercise1
!bambu icrc.c --top-fname=icrc1
```

Inspect the generated Verilog file in the explorer tab on the left (icrc1.v)

Executing (19s) Cell > system() > \_system\_compat() > \_run\_command() > \_monitor\_process() > \_poll\_process()

# Conclusion

- **FPGAs are very versatile and suitable for many markets**
- **Integrating HLS will improve productivity**
- **Raise the level of abstraction to develop rad-hard FPGA-based applications**
- **Raise the Technology Readiness Levels (TRL) of the Bambu HLS tool**



# Questions



# HERMES

HERMES PROJECT – H2020

Qualification of High-pErformance pRogrammable Microprocessor  
and dEvelopment of Software ecosystem

<https://www.hermes-h2020project.eu/>

<https://panda.dei.polimi.it>

<https://github.com/ferrandi/PandA-bambu>



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101004203