June 11, 2024
Recent Advances in European Space FPGAs: Technologies and Applications

High-Level Synthesis with Bambu: the HERMES Project Experience

Fabrizio Ferrandi

Politecnico di Milano, Italy,

Outline

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

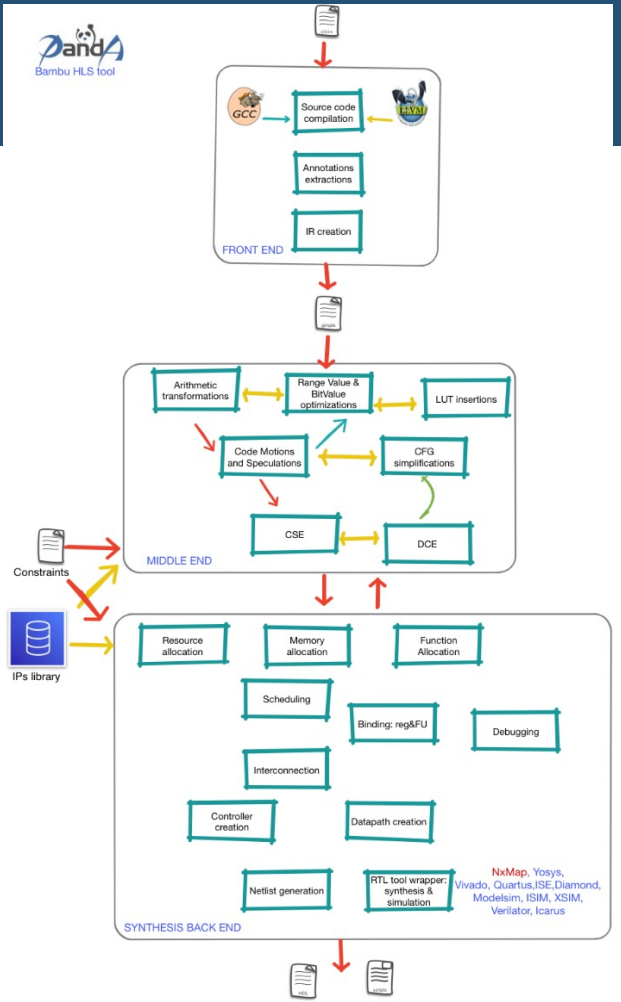
HERMES project

- **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

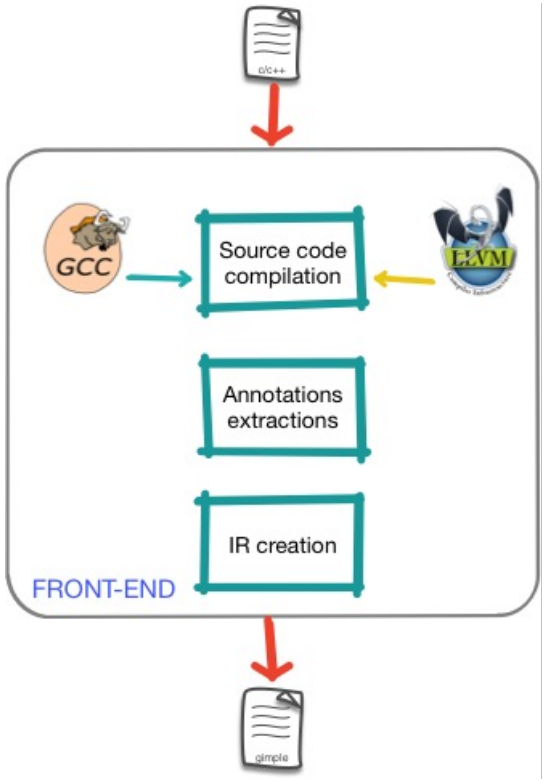
The increased performance offered by FPGAs is made available also to software developers that do not have hardware design expertise



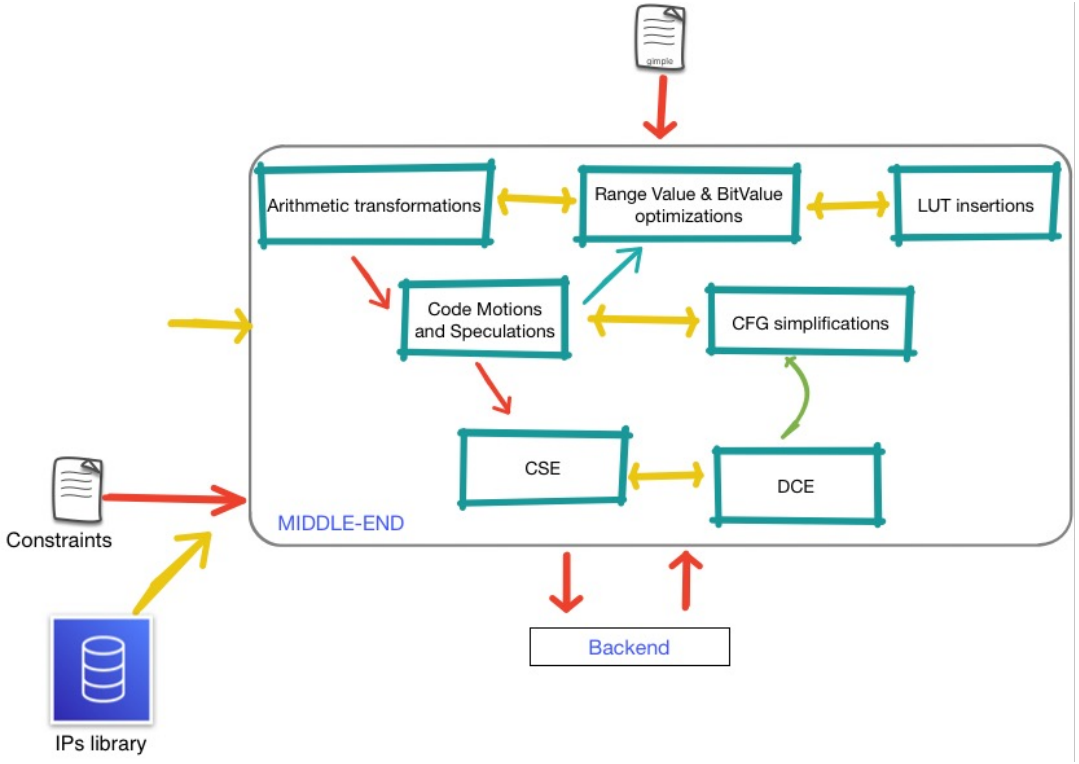
Bambu: an example of modern HLS tools

- **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)**

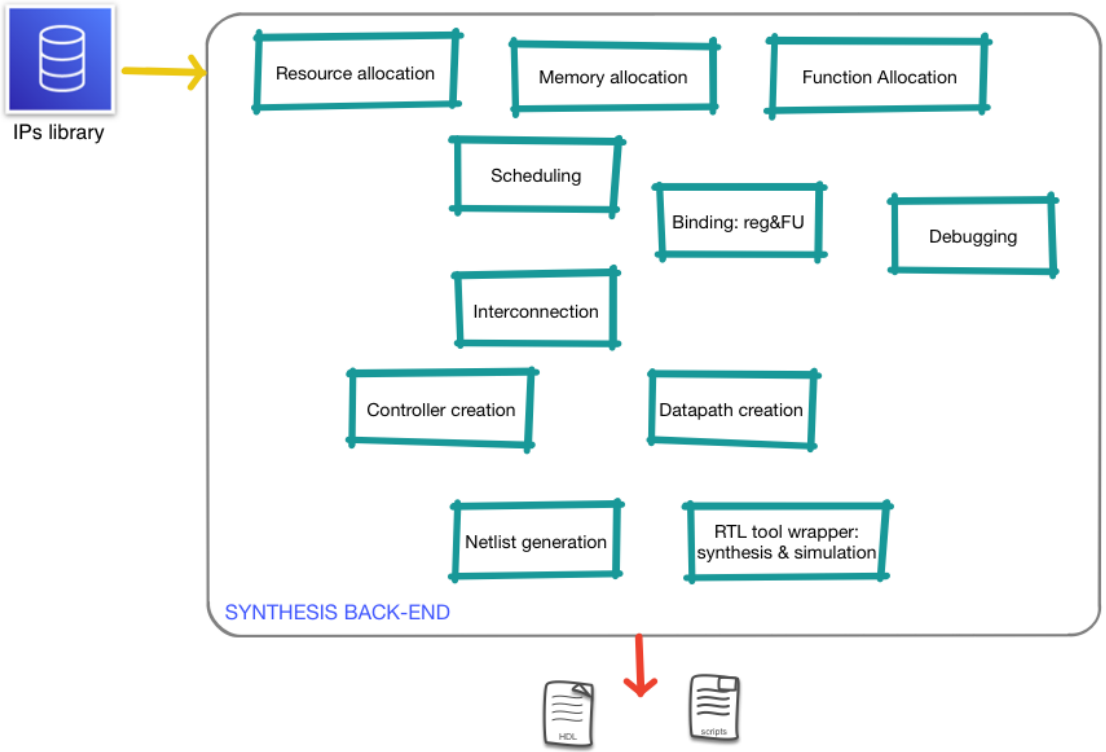
Bambu: front-end



Bambu: middle-end

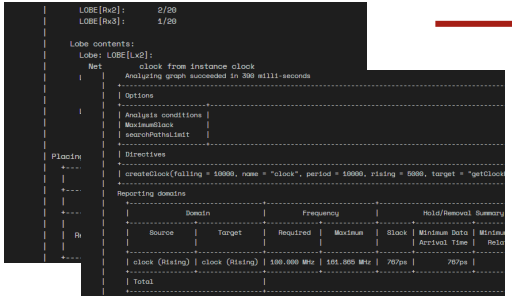


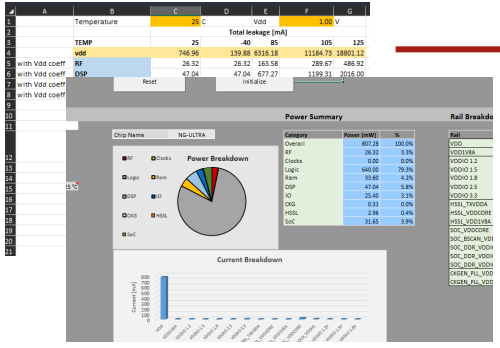
Bambu: back-end



Bambu – NX integration

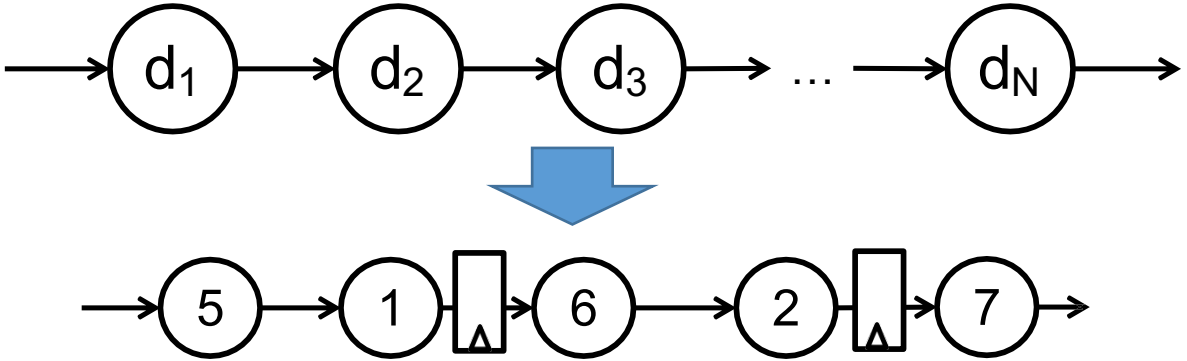
- Seamless integration between Bambu and Impulse from NX
  - automatic generation of backend synthesis scripts
- Bambu IP resource library characterized with respect to rad-hard NX FPGAs
  - Resource occupation and latency under different design constraints
  - Automatically performed with PandA Eucalyptus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

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

Interface pragmas

- **Very similar to the ones used by VitisHLS**
  - **No Protocol (standard parameter behavior)**
  - **FIFO, AXIS**
  - **Handshake**
  - **Array**
  - **Axi4-Master**
- **In many cases it is possible to use the same VitisHLS syntax. Not all the options are supported.**

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

- **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());
}
```

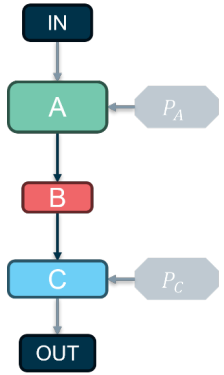
Bambu – extended features

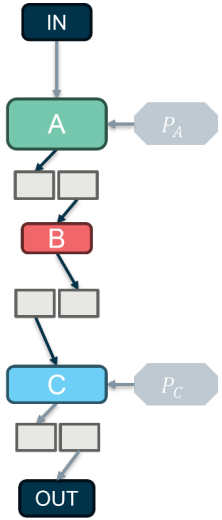
Dataflow support

```
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);
}
```



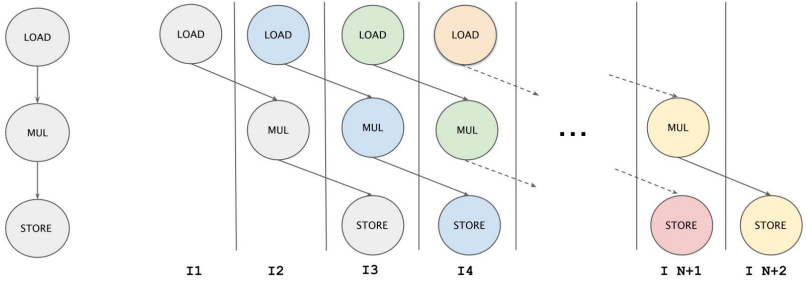


Dataflow current status

- ac_channel & hls_stream supported
- fifo depth could be controlled by pragmas
- Struct passed as template parameter supported
  - Synthesis efficiency could be improved
- Data field member support could be improved
- ac_int and ac_fixed supported
  - Synthesis efficiency comes when PandA-Bambu is used with Clang16

Bambu – extended features

- **Extended function pipelining support**
  - **User-defined initiation interval (II)**
  - **Optimized pipelining algorithm**



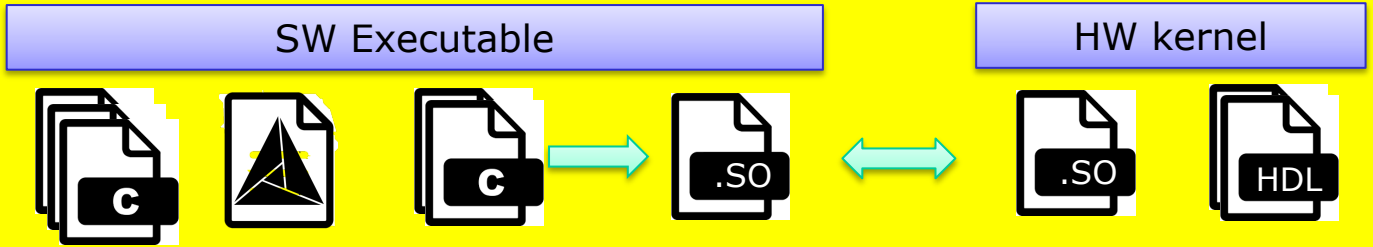
- **Loop pipelining overlaps execution of different iterations**
  - ►**Requires information about operation dependencies and available resources**
- **Latency reduction, less area overhead than loop unrolling**
- **Goal: initiation interval (II) = 1**

Polimi benchmark: Polybench kernels

- Suite of basic linear algebra operations commonly used to test polyhedral optimizations.
- Synthetized on a NanoXplore NG Ultra FPGA with target clock period of 20ns (50 MHz);
- Testing the loop pipelining and loop unrolling capability of Bambu HLS with 200 different experiments.
- 82% of the experiments reached the target frequency.

Bambu verification approach

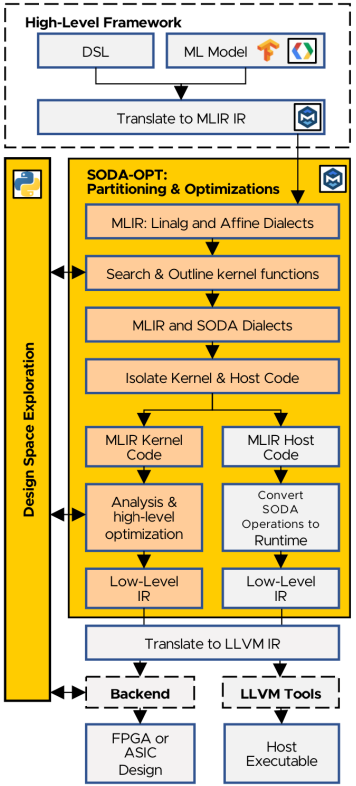
1. **Standard verification flow is now supported even by Bambu**
2. **Bambu additional verification approach**



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

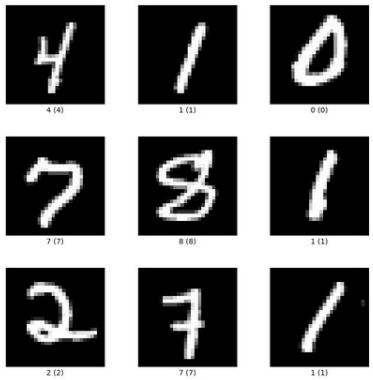
Integrating ML-design flow in Bambu

- **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)

MNIST example



input_1	input:	[(1, 28, 28)]
InputLayer	output:	[(1, 28, 28)]

↓

reshape	input:	(1, 28, 28)
Reshape	output:	(1, 28, 28, 1)

↓

conv2d	input:	(1, 28, 28, 1)
Conv2D	output:	(1, 26, 26, 12)

↓

max_pooling2d	input:	(1, 26, 26, 12)
MaxPooling2D	output:	(1, 13, 13, 12)

↓

flatten	input:	(1, 13, 13, 12)
Flatten	output:	(1, 2028)

↓

dense	input:	(1, 2028)
Dense	output:	(1, 10)

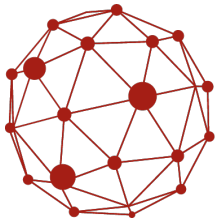
	NG-Ultra embedded
LUTS	4627
Registers	5714
Frequency	45.7 MHz
DSP	54
MEM	34
cycles	169,649

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

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 **H**igh-p**E**rformance p**R**ogrammable **M**icroprocessor and d**E**velopment of **S**oftware 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