Enhancing EDA Physical Synthesis Workflows with najaeda for the RISC-V Ecosystem

Christophe Alexandre *Noam Cohen*

keplertech.io

RISC-V enables open hardware, but EDA remains dominated by closed tools

Challenges in Scaling Open-Source EDA

- ❑ Limited access to modern process design kits (PDKs)
- ❑ Poor scalability for large, hierarchical designs
- ❑ Lack of advanced timing and physical closure tools
- ❑ Fragmented toolchains with inconsistent data exchange
- ❑ Reliance on outdated or Tcl-centric scripting interfaces

Open Source EDA Must Evolve to Meet the Demands of Next-Generation Silicon

- ❑ Support large, hierarchical netlists without flattening
- ❑ Enable fast, incremental design changes (ECO flows)
- ❑ Move from Tcl to Python-based, software-driven workflows
- ❑ Integrate with AI and data science tools (e.g., PyTorch, Pandas)
- ❑ Preserve design fidelity across the entire flow
- ❑ Provide abstractions for timing, placement, and routing

najaeda: Scalable Open EDA

What is najaeda ?

A Python-based framework for post-synthesis EDA, built for scalability and flexibility

- ❑ Install easily with '**pip install najaeda**'
- ❑ Access a Python-native API (no more Tcl!)
- ❑ Handle large, hierarchical netlists with high capacity
- ❑ Designed for ease of use and rapid prototyping
- ❑ **Built on optimized C++ data structures for performance**
- ❑ Preserves netlist fidelity — no flattening, no data loss

najaeda Python Script Examples

Load and Dump a Netlist

```
netlist.load_liberty(['NangateOpenCellLibrary_typical.lib'])
top = netlist.load_verilog('netlist.v')
top.dump_verilog('.', 'tinyrocket_naja.v')
```

Print Hierarchical Instance Tree

```
def print_netlist(instance):
    for child_instance in instance.get_child_instances():
        print(f"{child_instance}:{child_instance.get_model_name()}")
        print_netlist(child_instance)
```

Results and Perspectives

Optimizing RISC-V Netlists Using najaeda

Design	Characteristics	Initial Primitives	After najaeda Primitives	Logic Reduction
Basilisk	An end-to-end open-source, Linux-capable RISC-V System-on-Chip (SoC) developed by ETH Zürich and fabricated using IHP's open 130 nm technology.	950256	811302	-14,62%
blackparrot	A Linux-capable, cache-coherent, 64-bit RISC-V multicore processor designed to be the default open-source accelerator host	302530	250181	-17,30%
megaboom	One of the largest open-source RISC-V chips.	3871705	3042874	-21,40%

najaeda in Action: Real-World Applications

- ❑ **Hierarchical Netlist Optimization:** Efficiently optimize complex, multi-level netlists while preserving hierarchy and minimizing disruption to the original design.
- ❑ **FPGA Security Enhancements:** Embed cryptographic keys and implement protection mechanisms for tamper-resistant, secure FPGA designs.
- ❑ **Radiation Fault Tolerance:** Inject redundancy into registers and memory blocks to detect and correct soft errors caused by radiation.
- ❑ **Engineering Change Order (ECO) Automation:** Apply precise, incremental updates to existing designs using ECO flows without restarting the full synthesis cycle.

najaeda Python package

keplertech.io

contact@keplertech.io

najaeda GitHub page