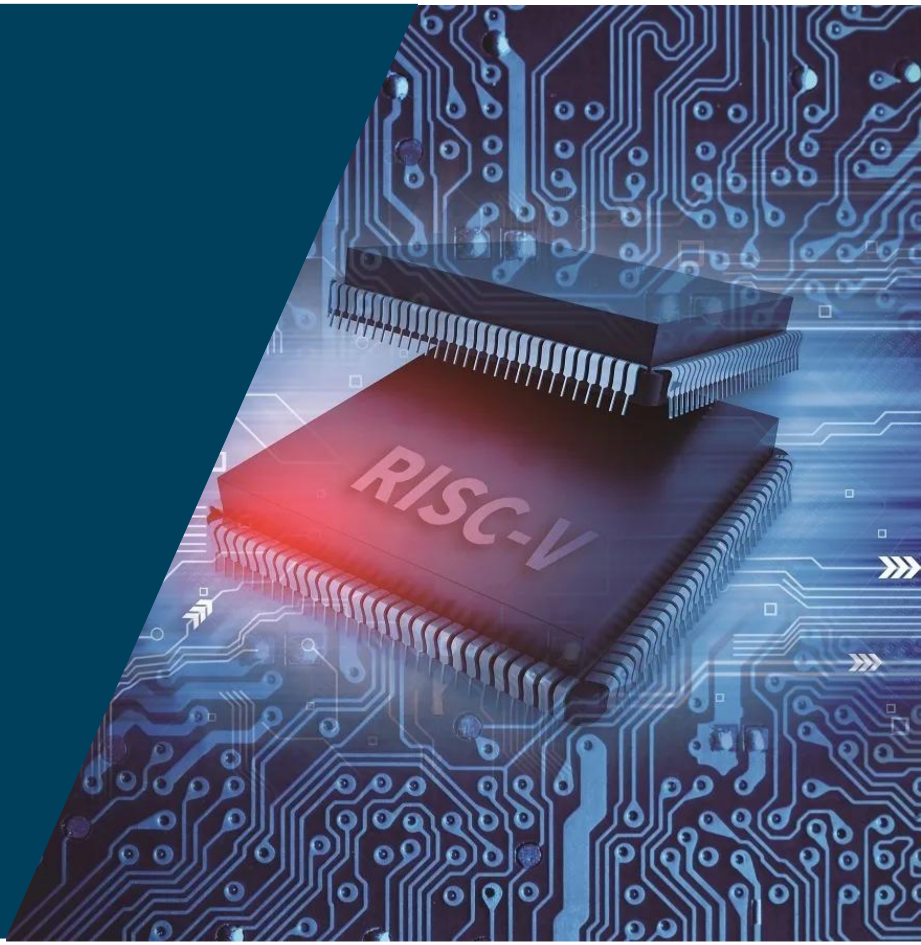
Validate and Implement a RISC-V core using AI

GDP 15

External Partner: Synopsys® Northern Europe Ltd

Team Members:
Vidushi Yaksh, Joseph Teh, Al Rawshan, William Ly, Y. Solomon Zhang, Cleon Kok

Academic Supervisor:
Mark Zwolinski



What is the Project About?

- *Aims of the project:*
- Validate and implement an open-source RISC-V core using Synopsys® tools and AI
- Assess the efficacy of DSO.ai by using it to optimize Power, Performance, and Area (PPA)

Verification

- UVM and Testbenches
- Synopsys VCS® Simulator
- Verdi® Debug Environment

Implementation

- Fusion Compiler™
- DSO.ai™

Project Specification

Verification

- Verifying and implementing the open source Bluespec MCU SoC
- Code and functional coverage analysis
- SAIF file generation for test with best code coverage

Implementation

- Technology: 14nm node using the SAED14nm cell library
- Floorplan Optimization:
  - Area: ≤ 65,000µm²
  - Core utilization: 70% - 80%
  - Core offset: 0.45nm
- Performance Goals:
  - Target frequency: 500MHz
  - Maximum power limit: 3.38mW
- Create Baseline using Fusion Compiler™
- Input Baseline into DSO.ai™ to maximize Power and Performance

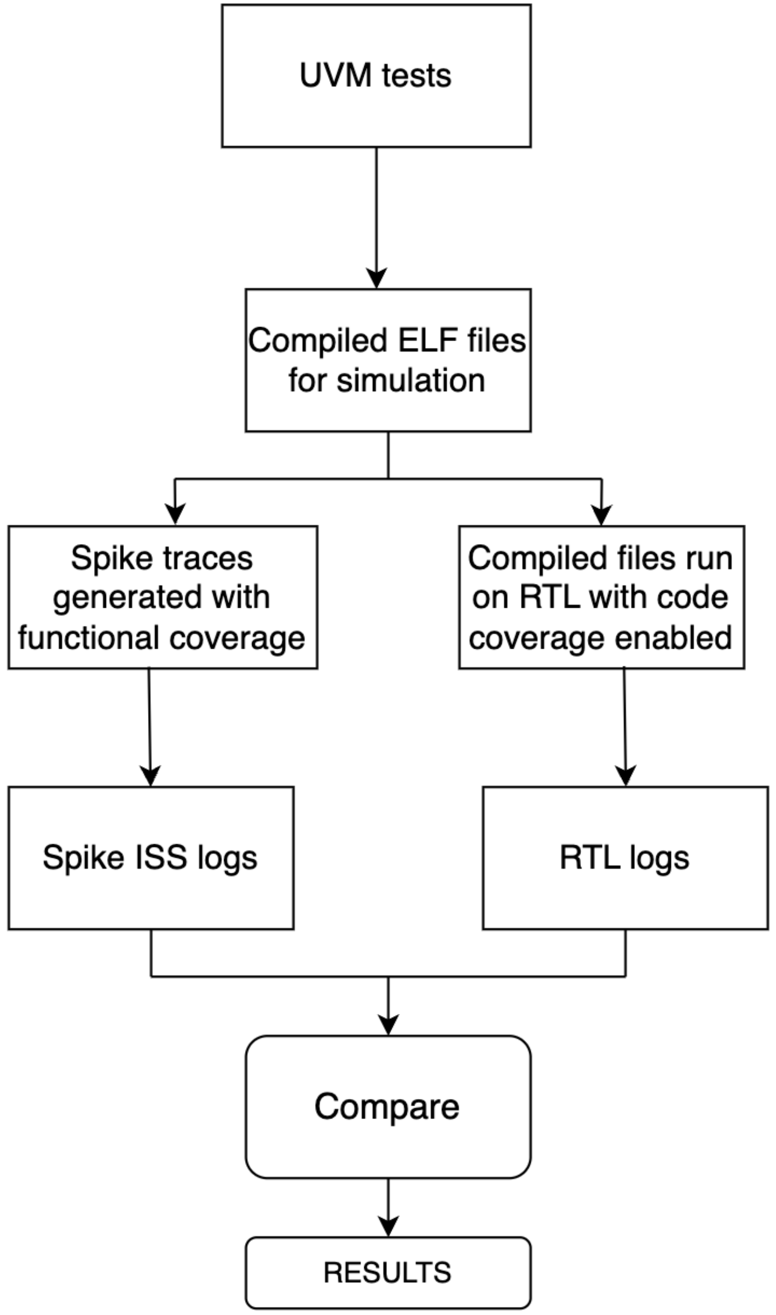
Verification

Verification

- The verification environment is spread across two testbenches

MCU Testbench (MCU TB)	Google Open-Source UVM based constraint random instruction generator (RISCV-DV TB)
RTL simulated using VCS®	Generate constrained random RISC-V instructions
Run bring-up test, ISA tests	Log comparison between SPIKE ISS and RTL
Code coverage is generated in this testbench	Functional coverage is generated in this testbench

Verification


UVM tests
Compiled ELF files for simulation
Spike traces generated with functional coverage
Compiled files run on RTL with code coverage enabled
Spike ISS logs
RTL logs
Compare
RESULTS

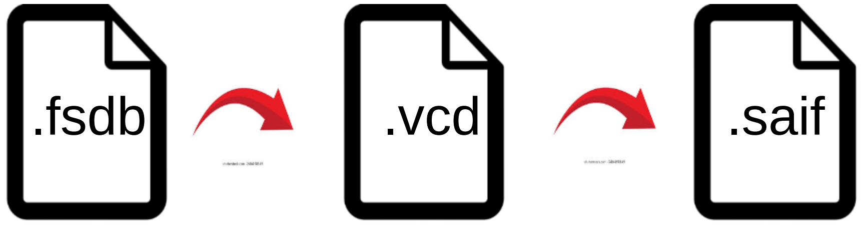
University of Southampton

SAIF File Generation

- Captures switching activities in the design during simulation, can be used to analyze power

Purpose:

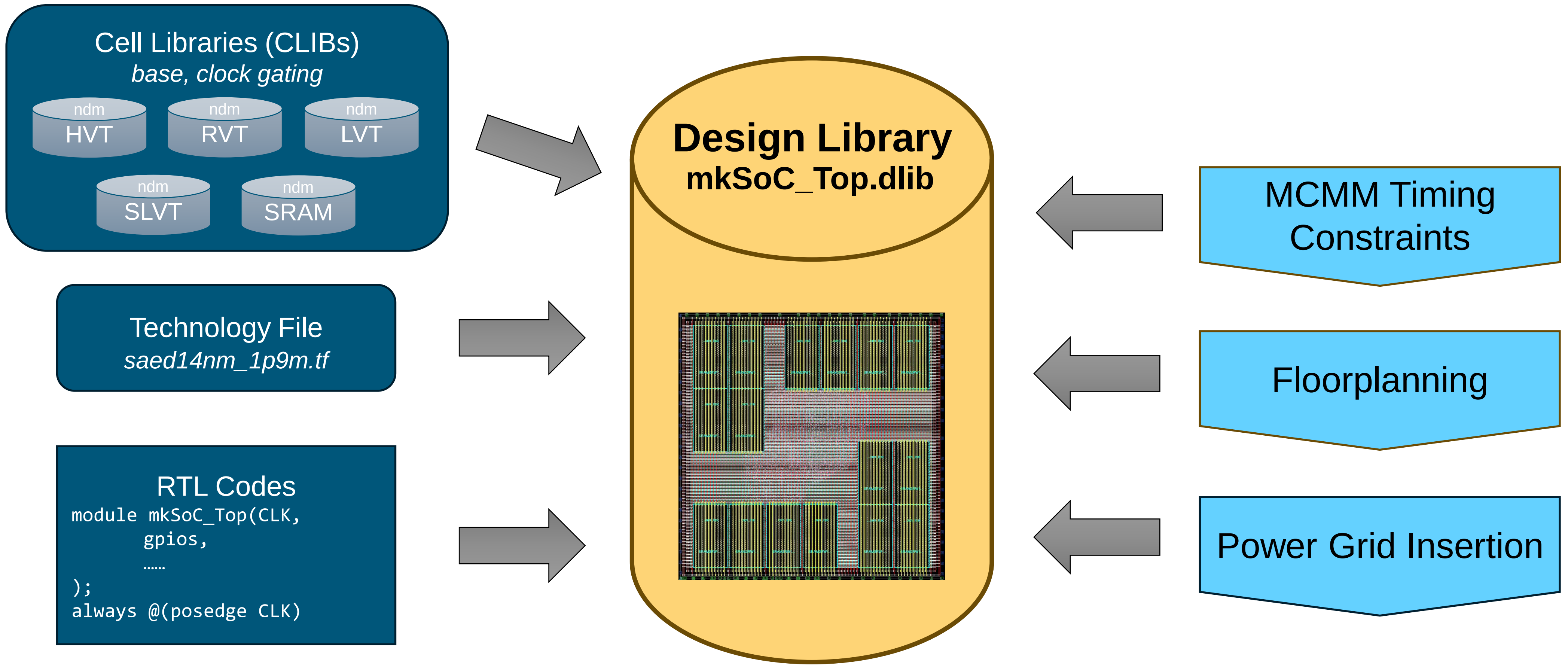
- Find the test with highest coverage
- Reflects switching activities that correlate with power usage
- Provides a more accurate power guideline for the implemented design



Implementation

Design Setup

Implementation

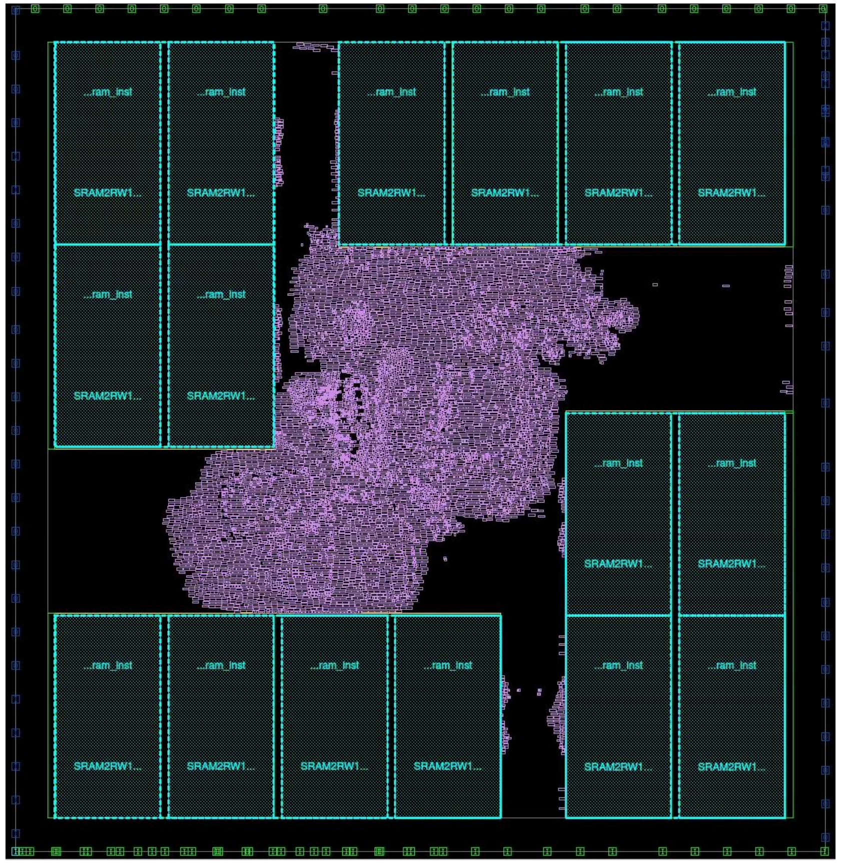


```
module mkSoC_Top(CLK,
       gpios,
       ……
);
always @(posedge CLK)
```

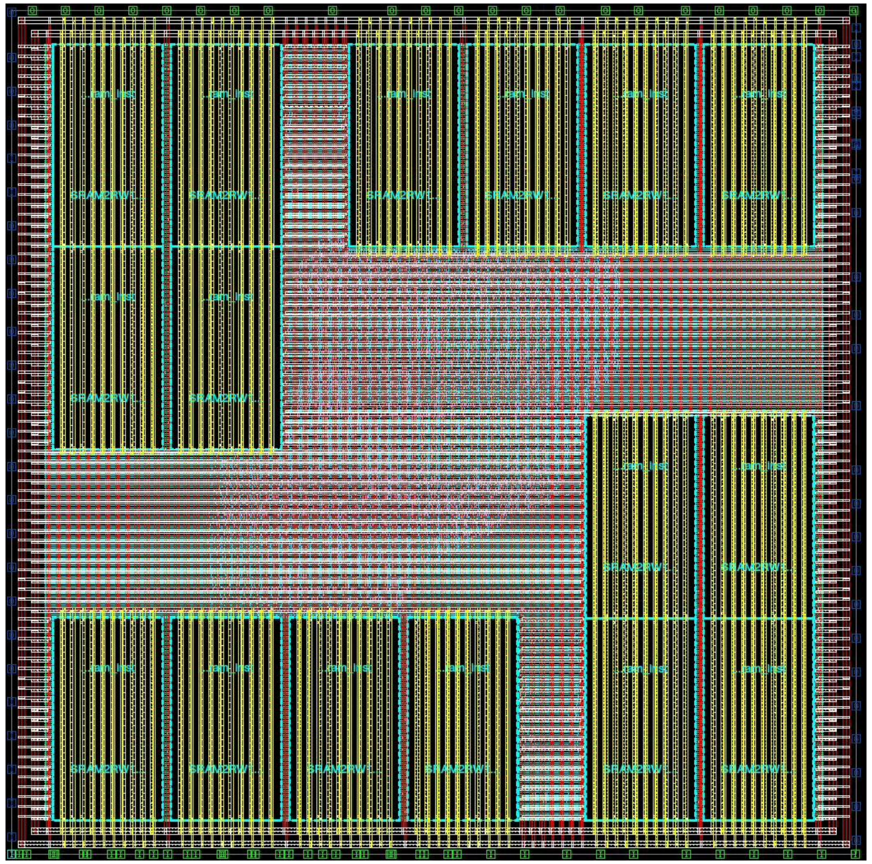
Floorplan & Power Grid Insertion

Implementation

Floorplan



Floorplan with Power Grid

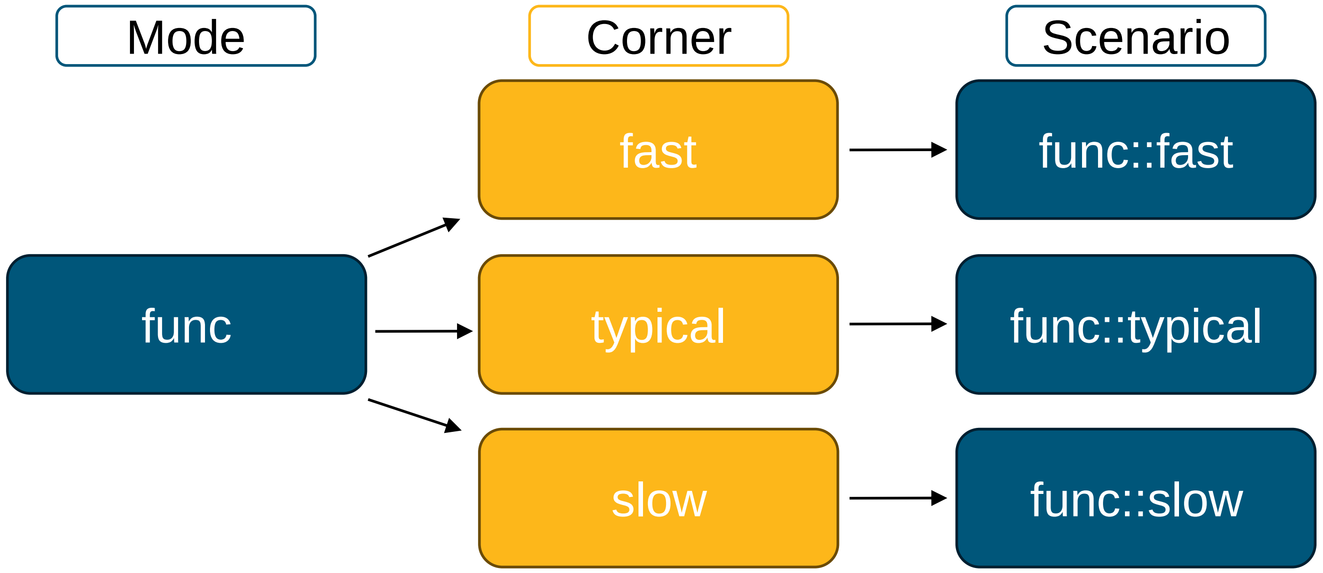


Area: 63,663.885 µm²
Core Utilization: 73.6%
Core Offset: 10 nm
Cache Size: 1 kB

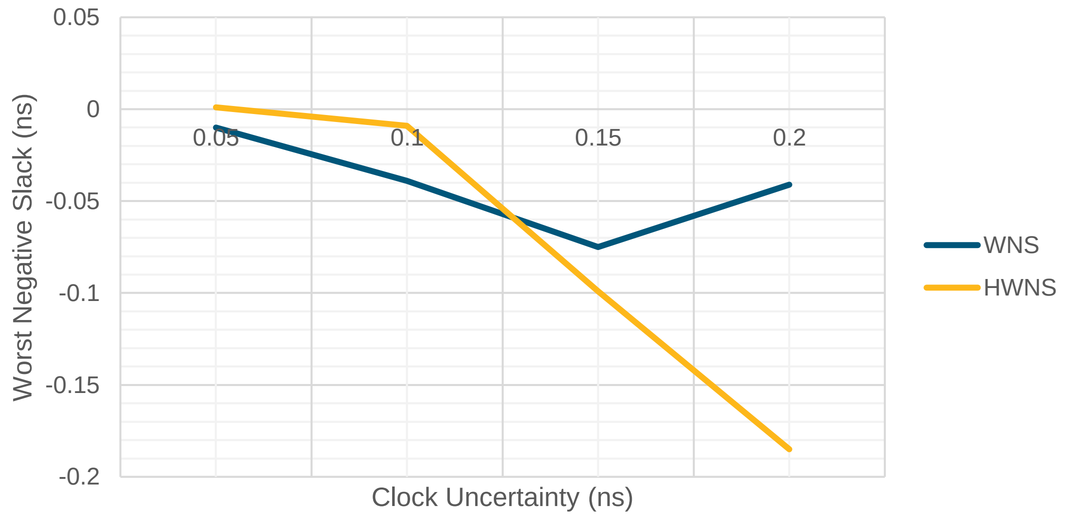
University of Southampton

Multi-Corner Multi-Mode Timing Constraints

Implementation



- Each corner must be constrained with a different Process, Voltage, and Temperature (PVT)
- Timing constraints modified iteratively using a script to optimise baseline's PPA



Design Space Optimization AI (DSO.ai™)

- Optimized a baseline's PPA by exploring unconventional search space solutions.
- Optimization strategies included optimizing each metric individually.
- Then explored a balanced trade-off between all metrics to find optimal solutions.

Why AI?

- Manual tuning is slow and limited
- Complex design space
- DSO.ai™ uses reinforcement learning
- Parallelized exploration saves time
- Objective-driven optimization

AI-Driven Design Exploration with Reinforcement Learning

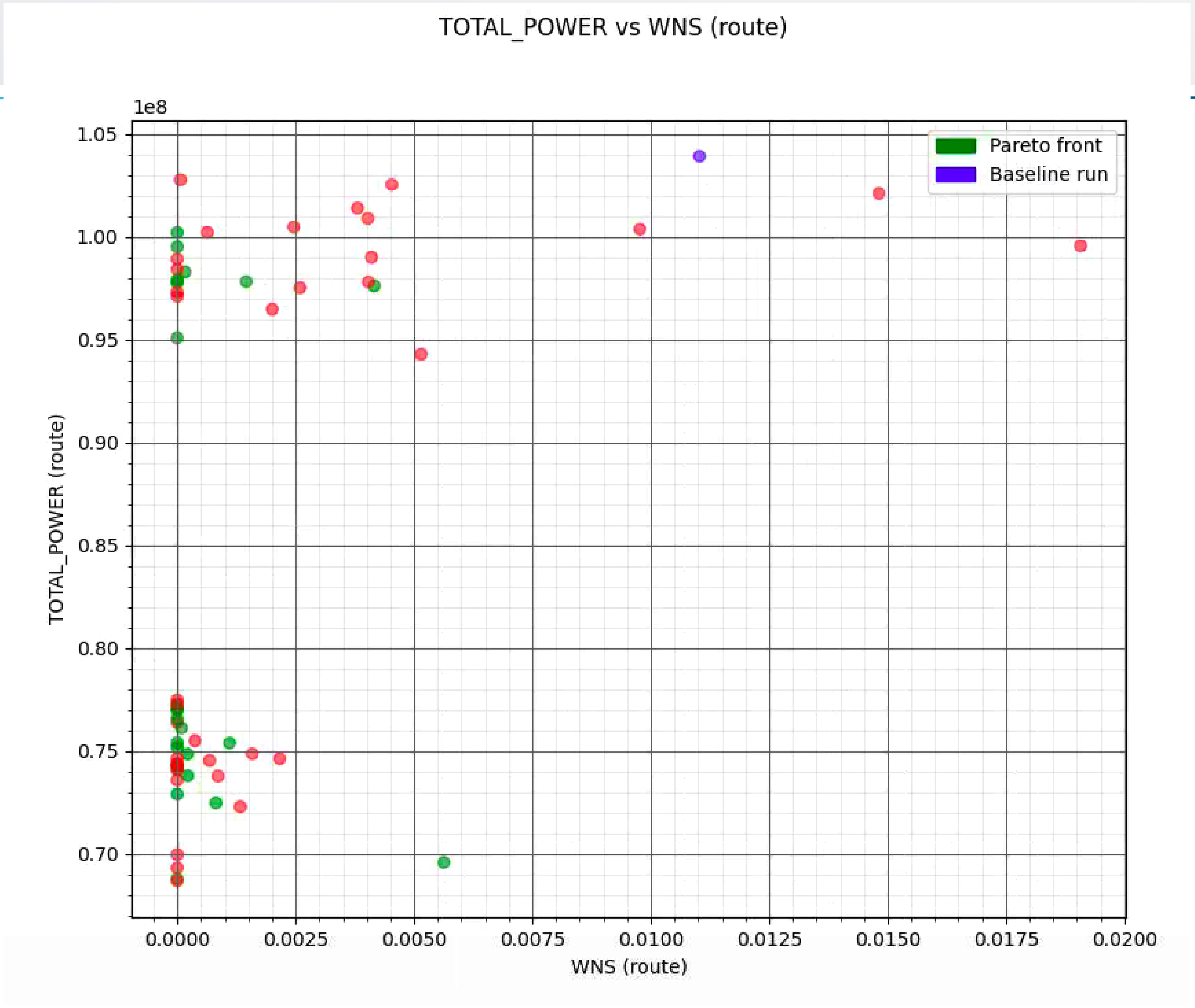
- DSO.ai™ uses reinforcement learning to explore the design space iteratively.
- Each design run is treated as an “action” with feedback in the form of PPA metrics.
- The AI learns from past results to improve its decisions over time.
- Best runs from one stage (slice) are used as inputs for the next stage (e.g., compile → clock → route).
- Efficiently avoids repeating poor configurations.

Prioritizing Metrics for Better AI Exploration

- AI needs clear goals to optimize — these are set through metrics.
- Common metrics: WNS, TNS, Total and Leakage Power, Area, etc.
- Two strategies used:
  - Pareto Optimization: Balances trade-offs across multiple metrics.
  - ADES (AI-Driven Efficiency Strategy): Assigns weights to focus on specific goals.
- Helps DSO.ai™ focus on what matters most for the design.
- Leads to faster convergence and more meaningful results.

Pareto Example

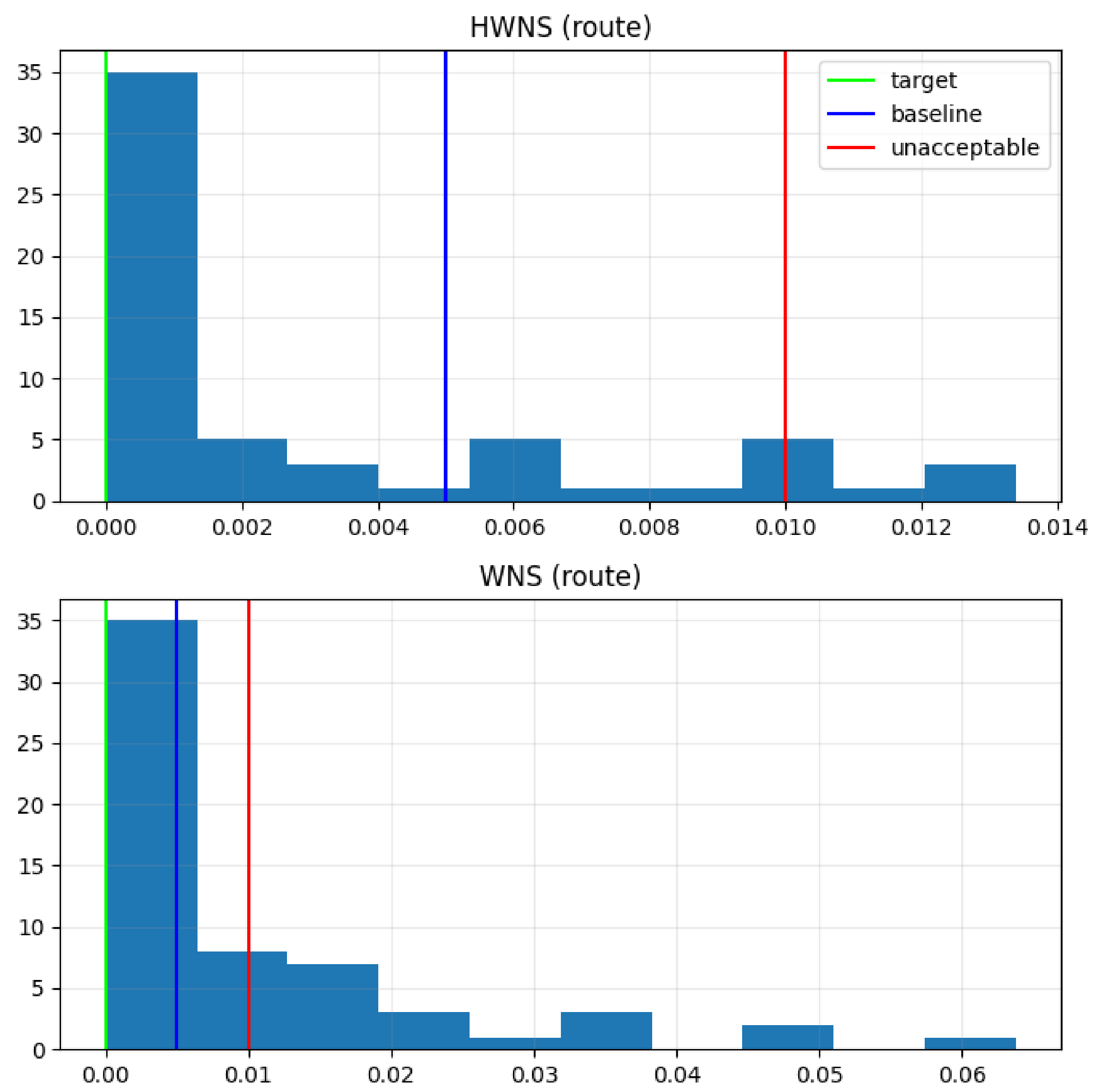
- Each dot = one DSO.ai™ run
- Green = Pareto-optimal trade-offs
- Purple = Baseline (higher power, worse timing)
- AI found lower-power, timing-closed designs
- Most runs cluster near WNS ≈ 0 ns



University of Southampton

ADES Example

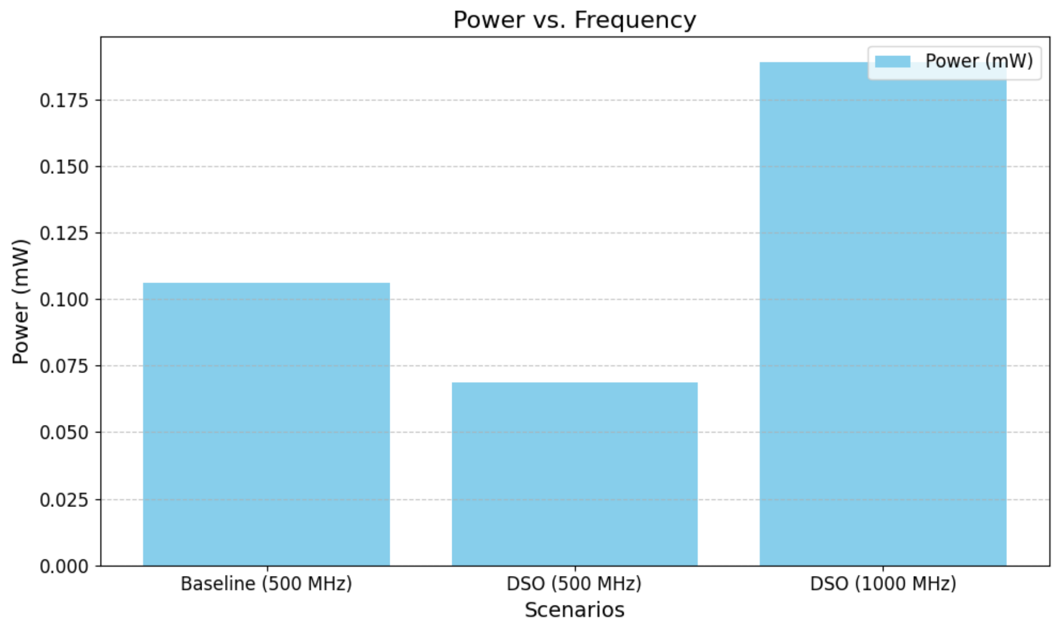
- Histogram of number of runs for each metric value
- Green = Target values for metrics
- Blue = Baseline metric value (worse than target value)
- AI optimized timing in expense of other metrics
- Most runs achieve lower WNS and HWNS ≈ 0.000 ns

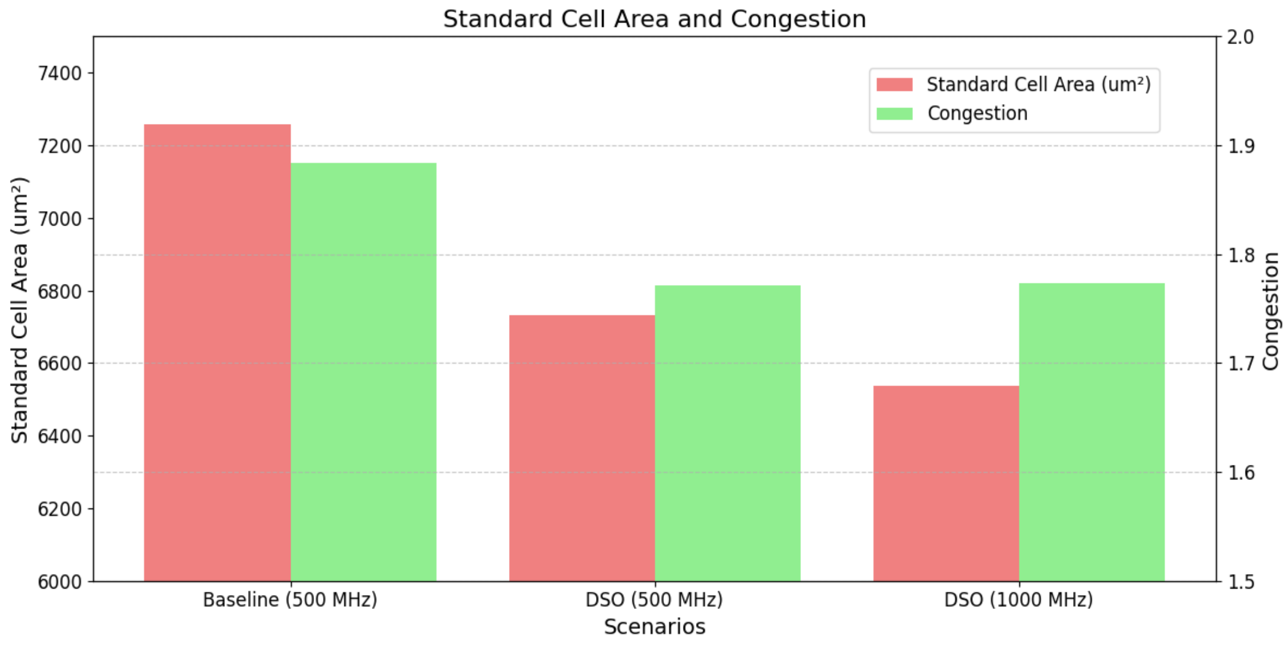


University of Southampton

AI Optimization vs Baseline Results

- DSO.ai™ produced 2 solutions with increased PPA compared to the Baseline
- Solution 1: Balanced PPA at 500MHz (35% power and 7% area improvement)
- Solution 2: 1GHz target frequency (9.9% area and 100% frequency improvement)





Results

Verification

- SAIF File generated for Fusion Compiler
- Code and functional coverage analysis done
- Optimized test suite with 63.06 CPU time

Implementation

- Frequency: 500MHz & 1GHz
- Power (SAIF included):
  - Baseline: 0.106mW
  - DSO.ai™ 500MHz: 0.0688mW
  - DSO.ai™ 1GHz: 0.189mW
- Floorplan:
  - Area: 63,663µm²
  - Core utilization: 70%
  - Core offset (Power Grid): 10nm

AI Usage Advantages and Disadvantages

Traditional Workflow	AI Usage (DSO.ai™)
• Requires a large team of skilled engineers to fine-tune performance • Engineering time can be high as projects can span months • Requires less computational resources to run	• A small team of less skilled engineers can fine-tune performance effectively • Achieve competitive outcomes with less resources in a shorter amount of time • Requires more computational resources to run

Conclusion

- AI usage enabled more effective performance tuning in a shorter time-period.
- Optimal baseline achieved within 6 - 7 weeks.
- 2 improved designs using DSO.ai™ achieved within 2 weeks.
- Only 3 students needed to improve PPA using DSO.ai™.
- More time allocated to prioritize power grid insertion due to time savings.

Thank you