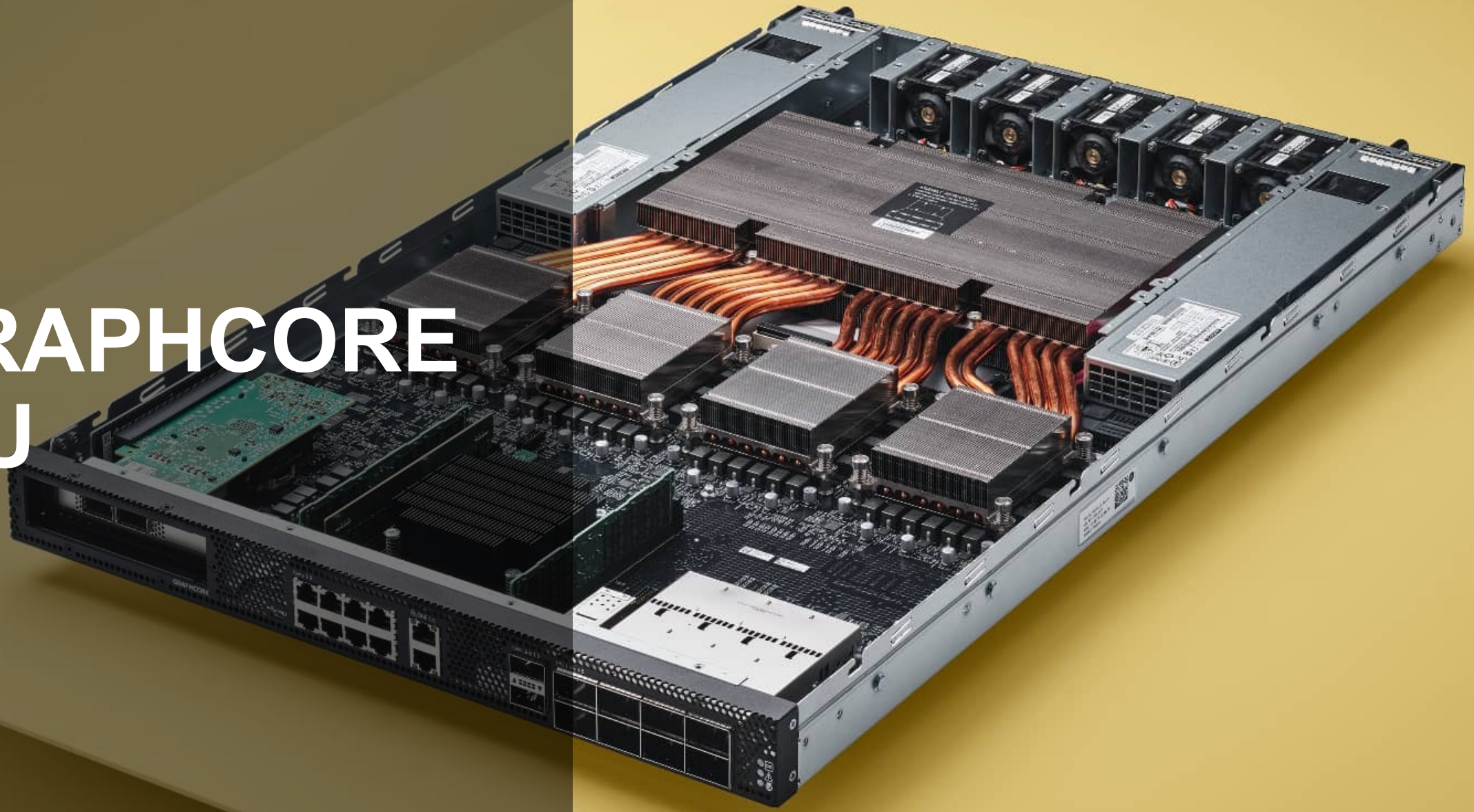
JUMPING IN THE DEEP END

A GRADUATE'S PERSPECTIVE ON VERIFYING AN ADVANCED AI CHIP

GRAPHCORE

GRAPHCORE IPU



GRAPHCORE IPU

IPU-Tiles™

1472 independent IPU-Tiles™ each with an IPU-Core™ and In-Processor-Memory™

IPU-Core™

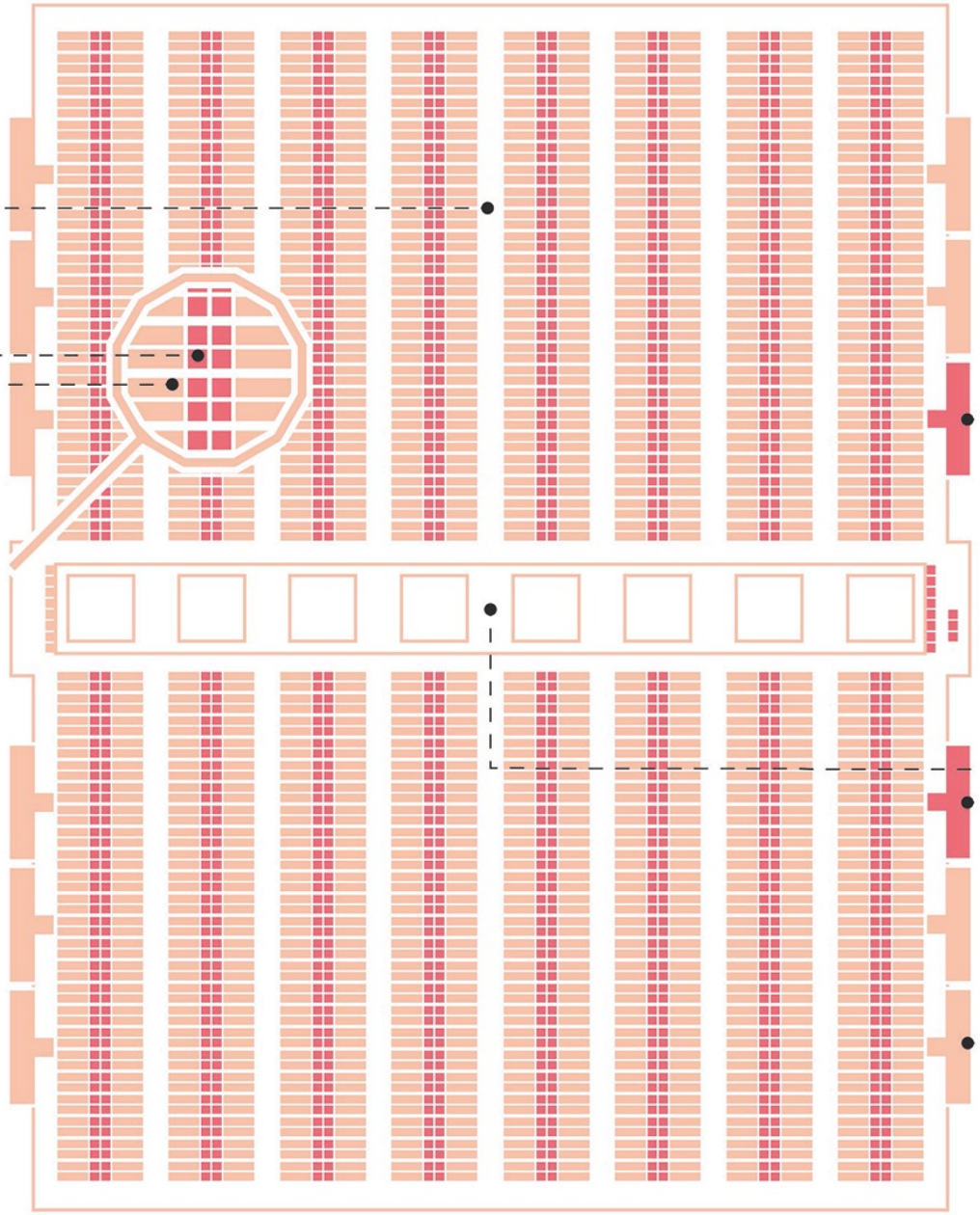
1472 independent IPU-Core™

8832 independent program threads executing in parallel

In-Processor-Memory™

900MB In-Processor-Memory™ per IPU

47.5TB/s memory bandwidth per IPU



IPU-Exchange™

8 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern

PCIe

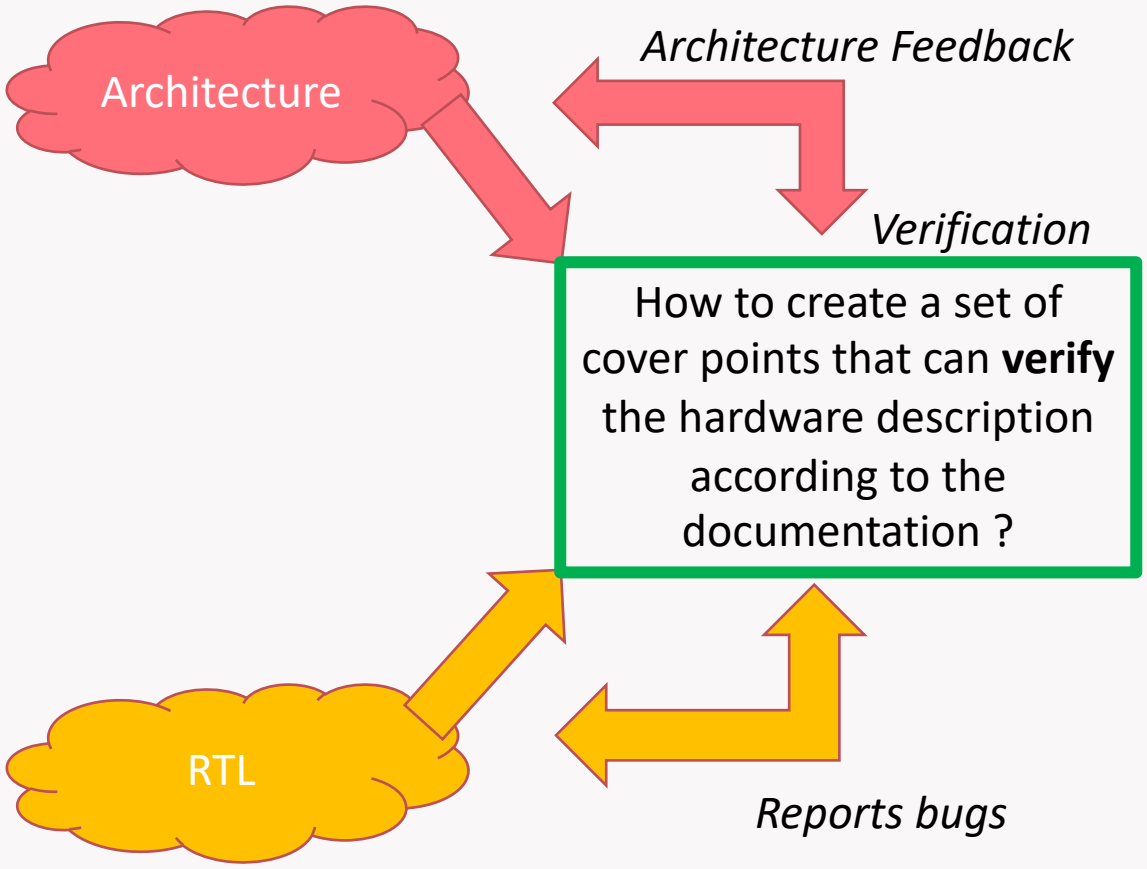
PCI Gen4 x16
64 GB/s bidirectional bandwidth to host

IPU-Links™

10 x IPU-Links,
320GB/s chip to chip bandwidth

GRADUATE VERIFICATION EXPERIENCE

GRADUATE EXPERIENCE – LOGAN (CORE VERIF)

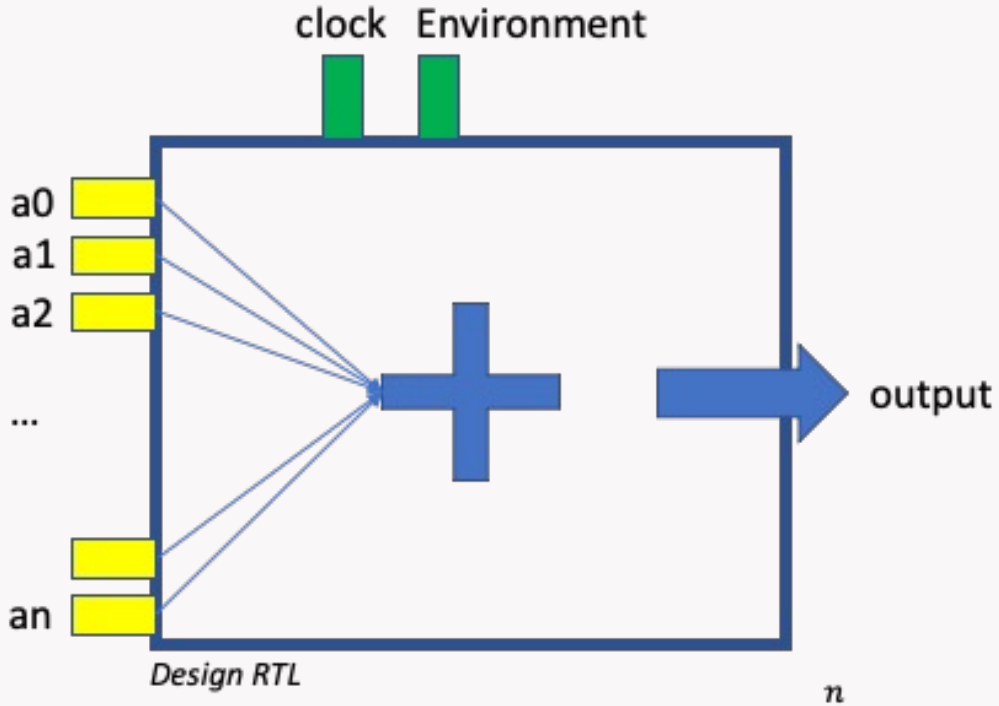


1. You will get a full <u>understanding of the architecture</u>. You will be able to find possible improvements and to give some feedback.

2. You will have a good <u>understanding of the hardware design</u> to create specific verification patterns.

3. You will learn to find a subset of the CPU state space that is optimal for verification which means <u>not computationally expensive</u> and that ensures hardware <u>high reliability</u>.

VERIFICATION CHALLENGE



$$f_{model}\ (a_0, a_1, \dots, a_n, Env, cycle) = \sum_{i=0}^{n} (a_i)$$

Architecture model – Mathematical Model

What about brute force ?

- If n=1 (a0, a1), we have $2^{64} \sim 10^{19}$ combinations for 32-bits inputs
- AMD RADEON – 8 TFLOPS $\sim 10^{13}$ op/s
- We would need 10^{6} sec to compute everything which means 1,7 week

This is a lower bound !

What is behind the Environment variable ?

What about the mathematical model ?

- Can we rely on the addition done on the AMD core ?

Conclusion

We need to find a better way !

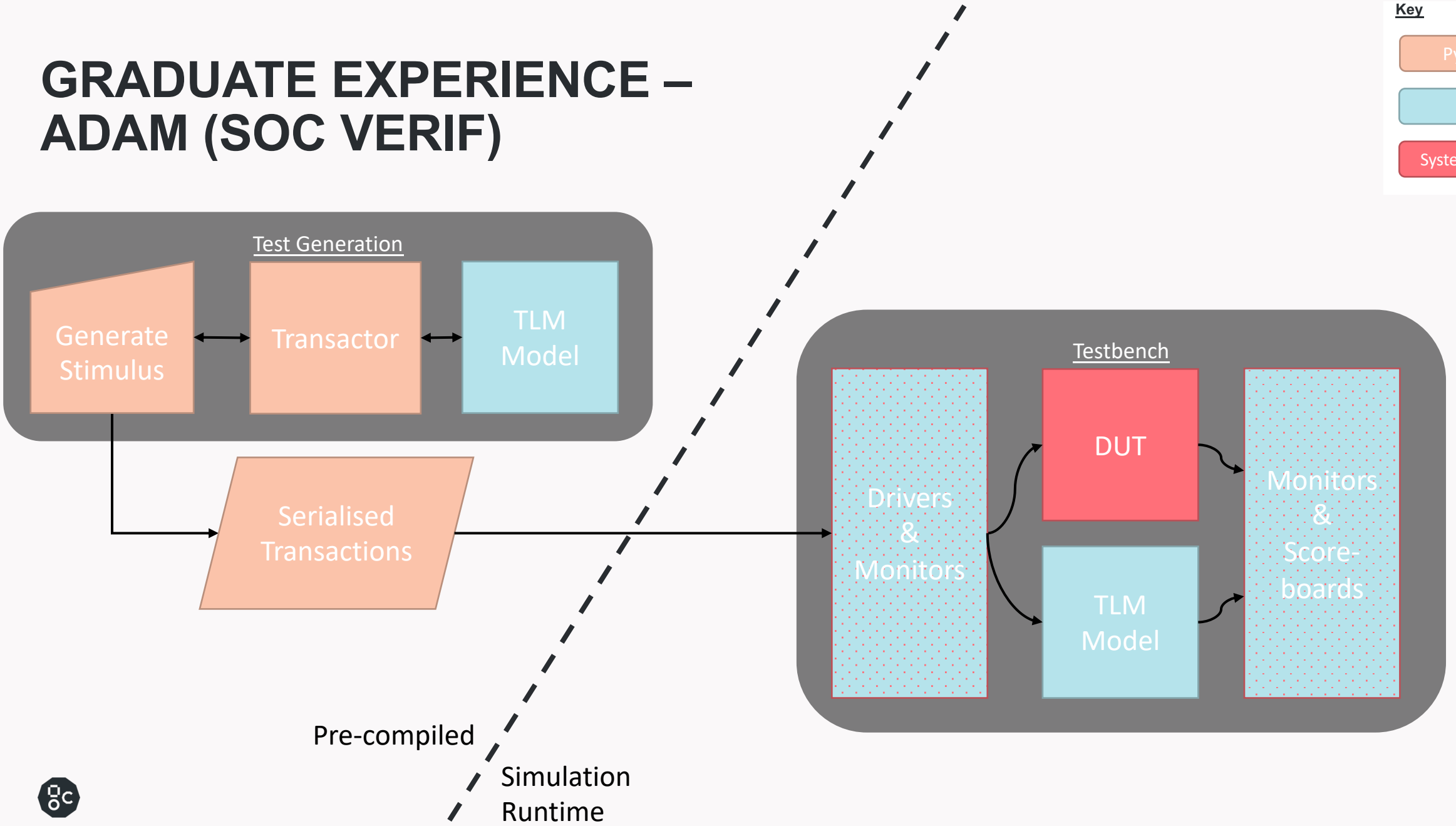
Can we rely on the following property ?

$$f_{model}\ (a_0, a_1, \dots, a_n, Env, cycle) = f_{model}\ (a_1, a_0, \dots, a_n, Env, cycle)$$

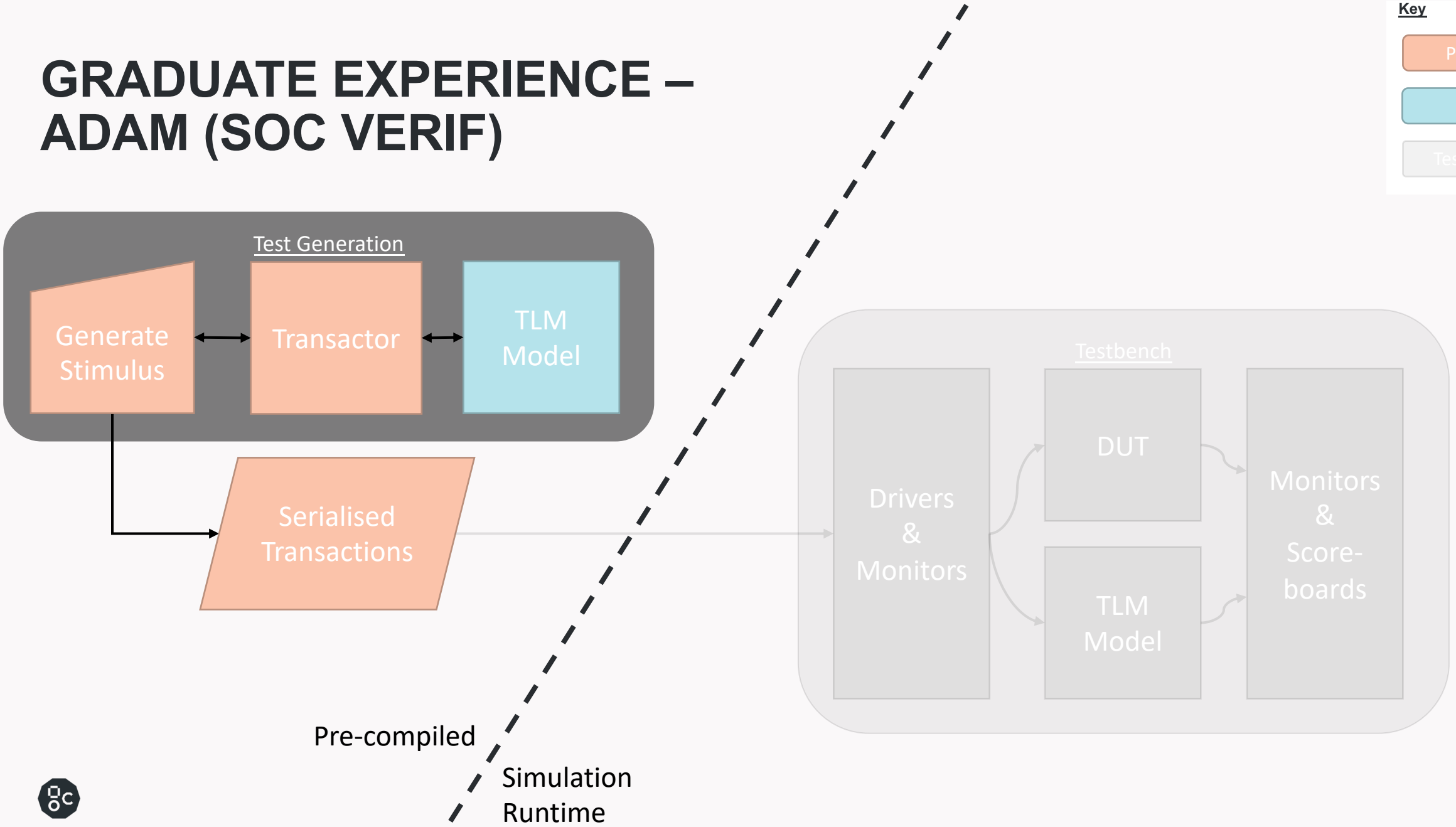
GRADUATE EXPERIENCE - ADAM (SOC VERIF)

- Joined Graphcore's Physical Design team in 2020
- Moved into SoC Verification in 2021
- Enjoying using Graphcore's verification methodology
- The verification work can feel very "software-like"
  - Exposure to Python, C++, SystemVerilog, …
  - High-level object-oriented programming
  - No more Tcl scripting 🎉🎉

GRADUATE EXPERIENCE – ADAM (SOC VERIF)

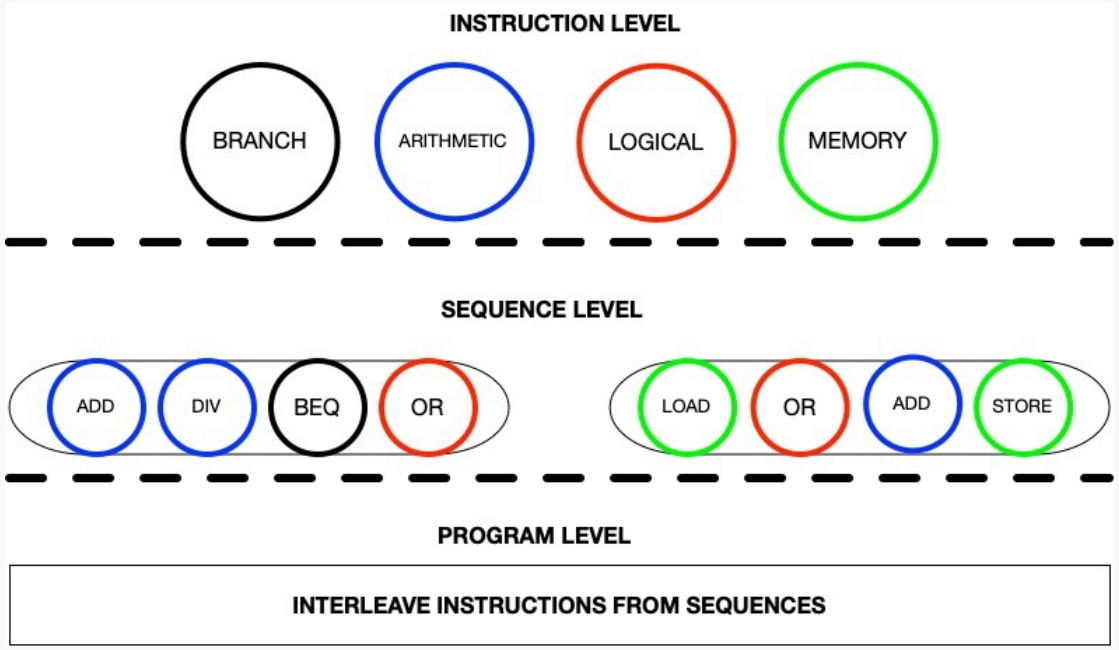


GRADUATE EXPERIENCE – ADAM (SOC VERIF)



GRADUATE EXPERIENCE – MIHAI (CORE VERIF)

INSTRUCTIONS GENERATOR

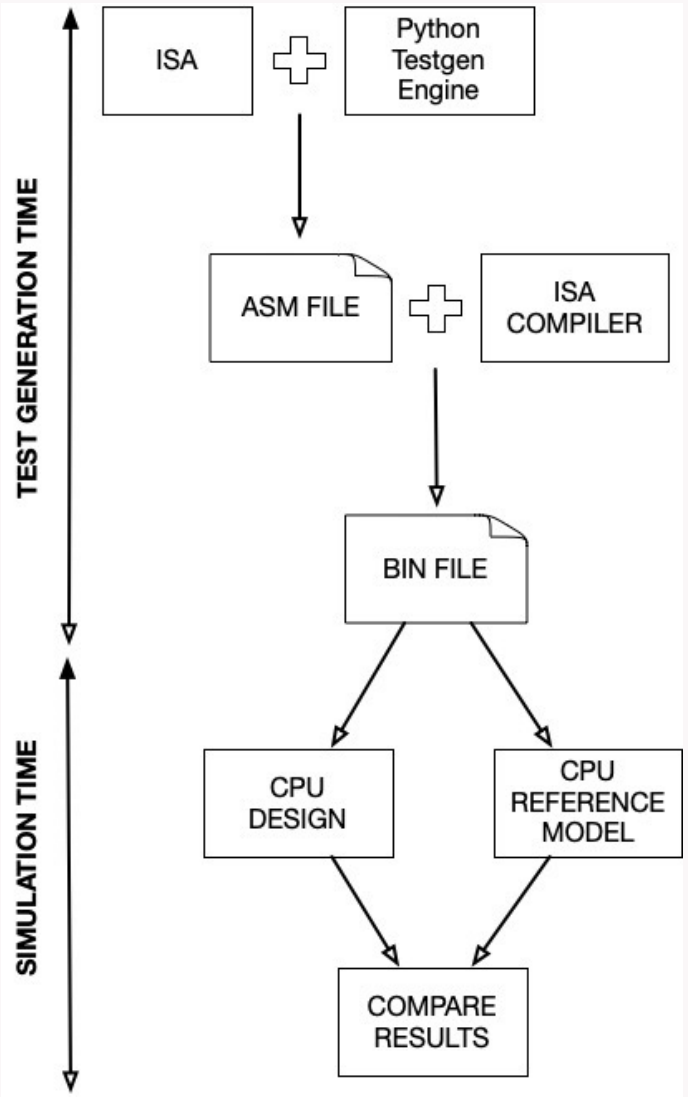


Randomness

- operands and immediate values

- different instructions order

- different CSR values, BOOT options, interrupts, debug interface



GRADUATE EXPERIENCE – MIHAI (CORE VERIF)

Chip roadmap
-> from design spec to product

Exposure to different EDA tools

WHY VERIF AT GRAPHCORE

Expand knowledge in
Logical, Physical, DFT

Tasks diversity which requires:
Databases, Regular expressions,
Power Analysis, Design Patterns,
Multithreading Programming

GRADUATE SOCIALISING

Register for future graduate roles and internships

WHERE TO FIND US

https://www.graphcore.ai/early-careers

students@graphcore.ai

https://www.linkedin.com/company/graphcore

https://www.youtube.com/c/Graphcore/videos

THANK YOU

Adam, Logan, Mihai
students@graphcore.ai