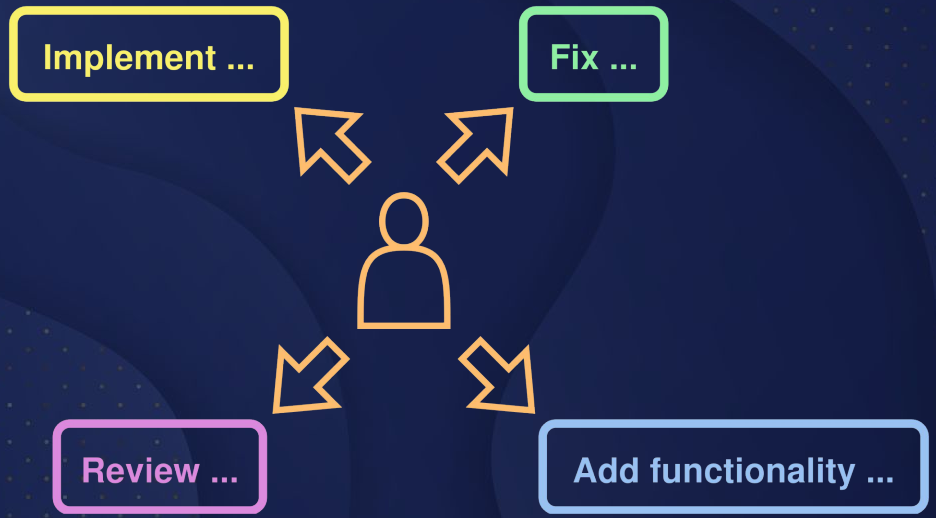
LLM-based Functional Coverage Generation and Auto-Evaluation Framework

Ján Labuda, Marcela Zachariášová, Zdeněk Matěj

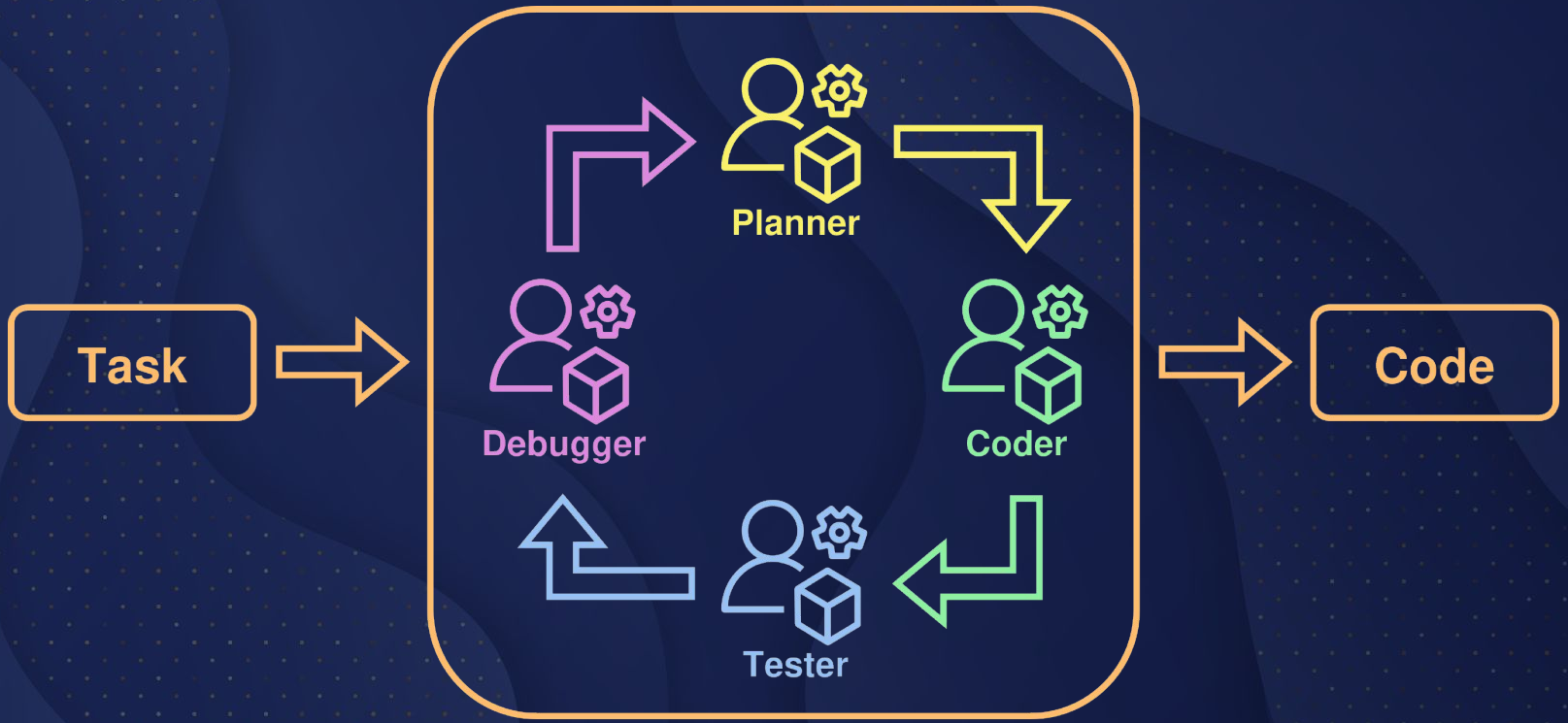
accellera SYSTEMS INITIATIVE

Daily interaction with LLMs

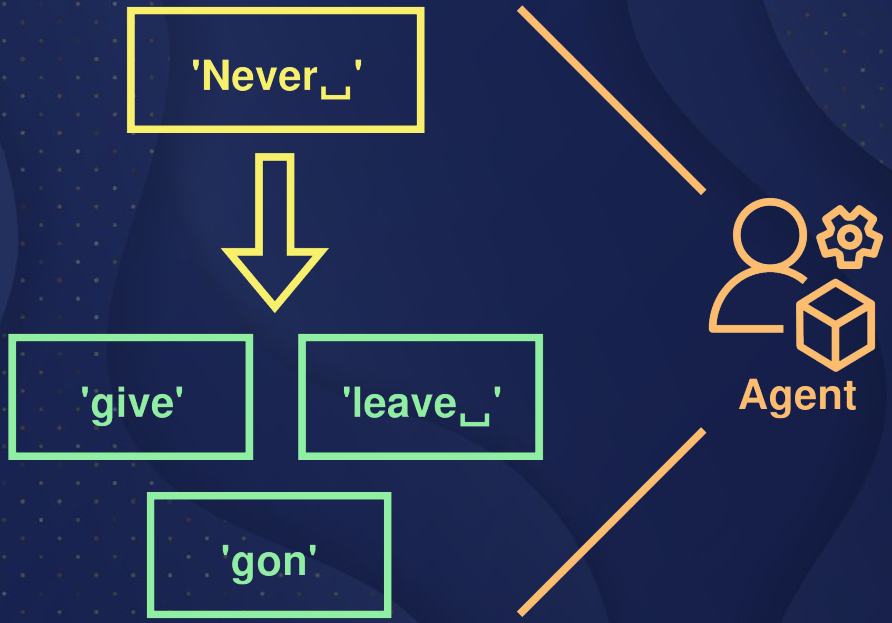
- Access to almost any solution from the Internet within seconds.
- How often have you argued with an LLM?



Raise of Agentic Systems



Deconstruction of Agentic Systems



Deconstruction of Agentic Systems

AI driven Chip Design

- Used in random stimuli generation, floor planning, and debug.
- Lacking a large quantity of good quality data to train LLMs.
- Application of agentic systems is still under the research.

AI driven Chip Design

DvCon USA 2025: Configurable Graph-Based Task Solving with the Marco Multi-AI Agent Framework for Chip Design

AI driven Chip Design

DvCon USA 2025: Configurable Graph-Based Task Solving with the Marco Multi-AI Agent Framework for Chip Design

NVIDIA: Configurable Graph-Based Task Solving with the Marco Multi-AI Agent Framework for Chip Design

AI driven Chip Design

DvCon USA 2025: Configurable Graph-Based Task Solving with the Marco Multi-AI Agent Framework for Chip Design

DeepMind: AlphaEvolve: A Gemini-powered coding agent for designing advanced algorithms

NVIDIA: Configurable Graph-Based Task Solving with the Marco Multi-AI Agent Framework for Chip Design

LLMs and Functional Verification

- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.

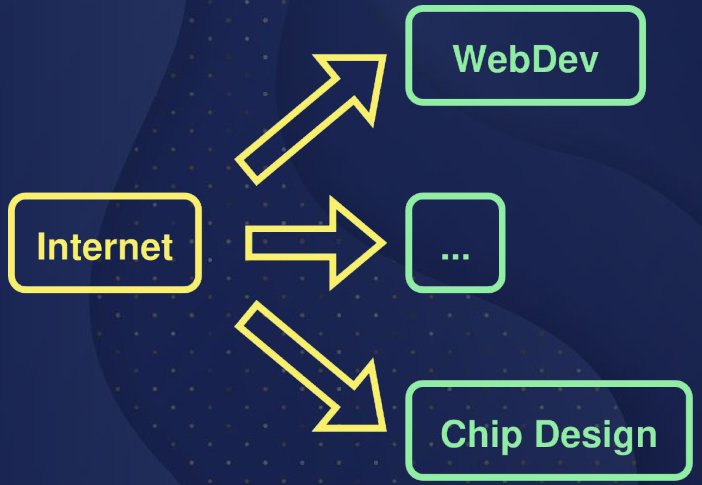
LLMs and Functional Verification

- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.

Internet

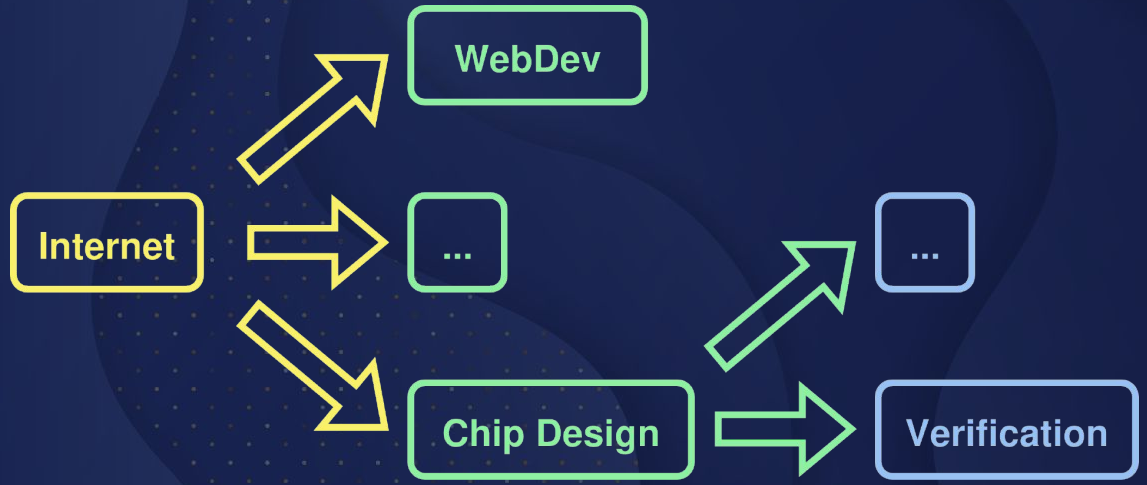
LLMs and Functional Verification

- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.



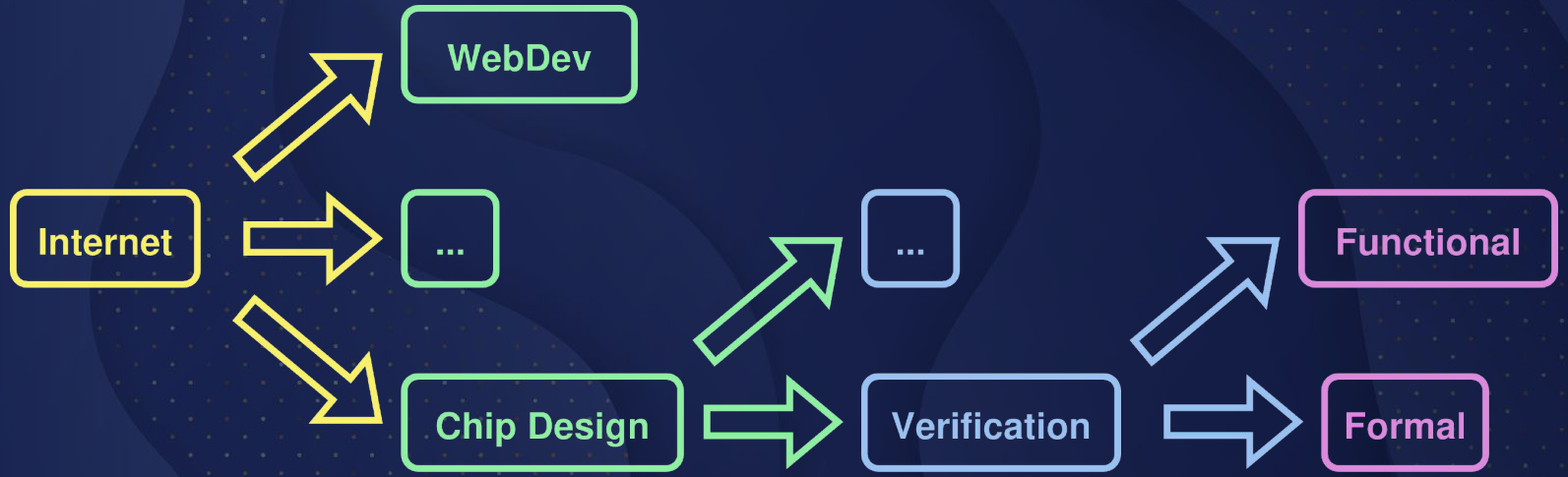
LLMs and Functional Verification

- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.



LLMs and Functional Verification

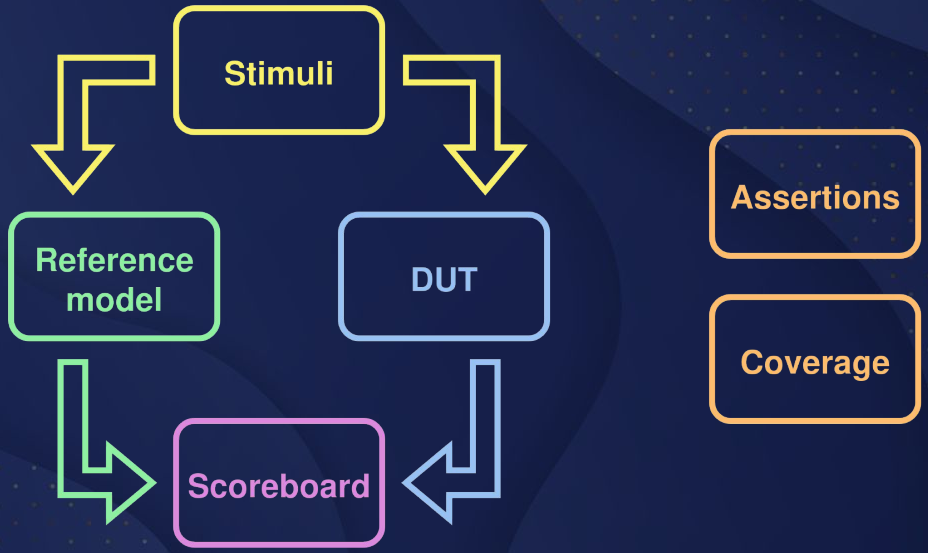
- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.



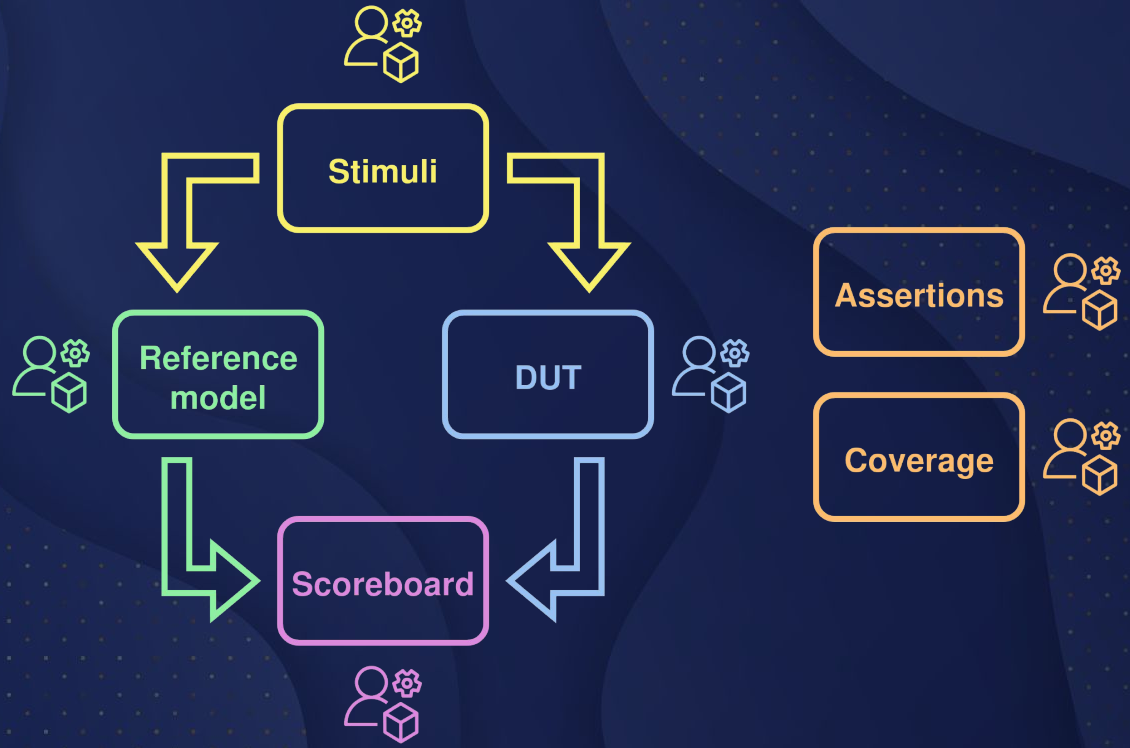
LLMs and Functional Verification

- Topic still in the research.
- EDA vendors proposing new agentic systems.
- Initial experiments shown that LLMs struggle with UVM testbenches.

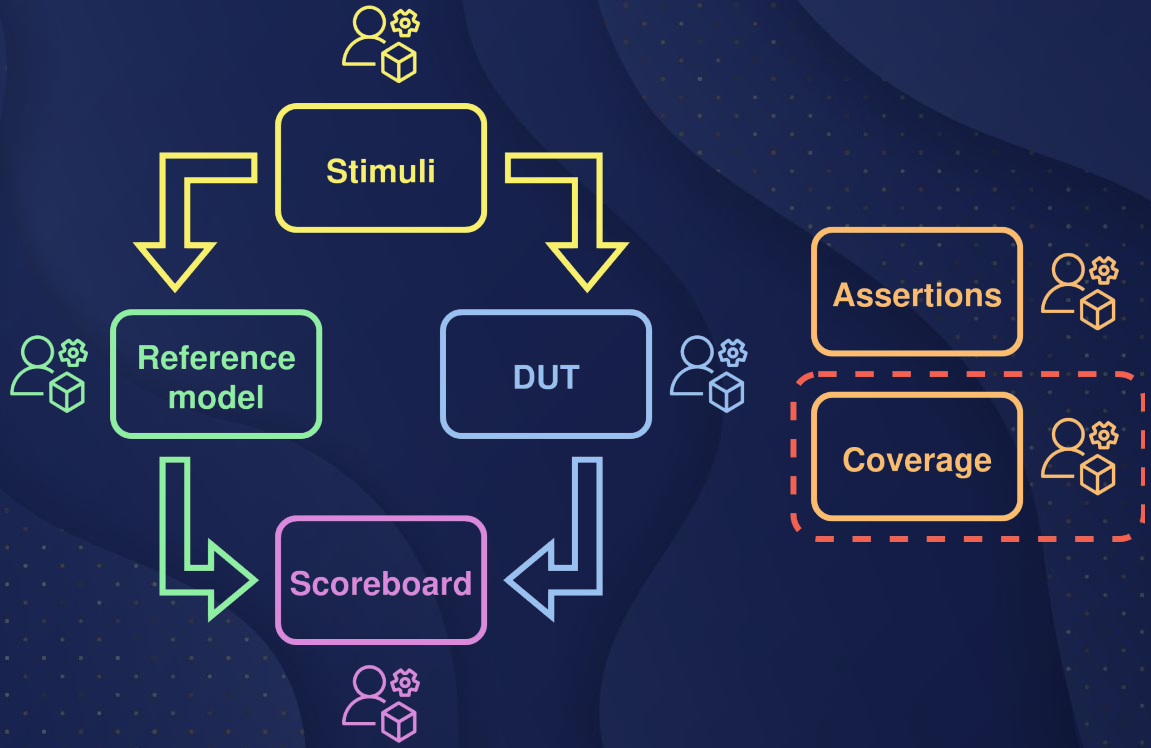
Decomposition of Functional Verification



Decomposition of Functional Verification



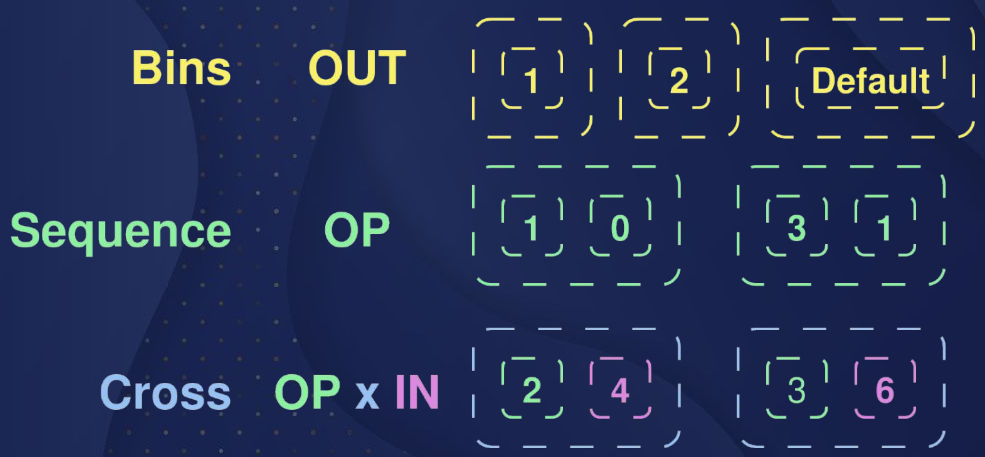
Decomposition of Functional Verification

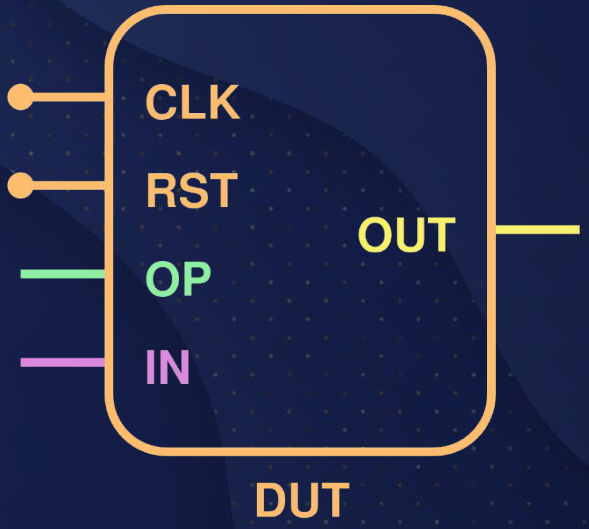


Why Functional Coverage?

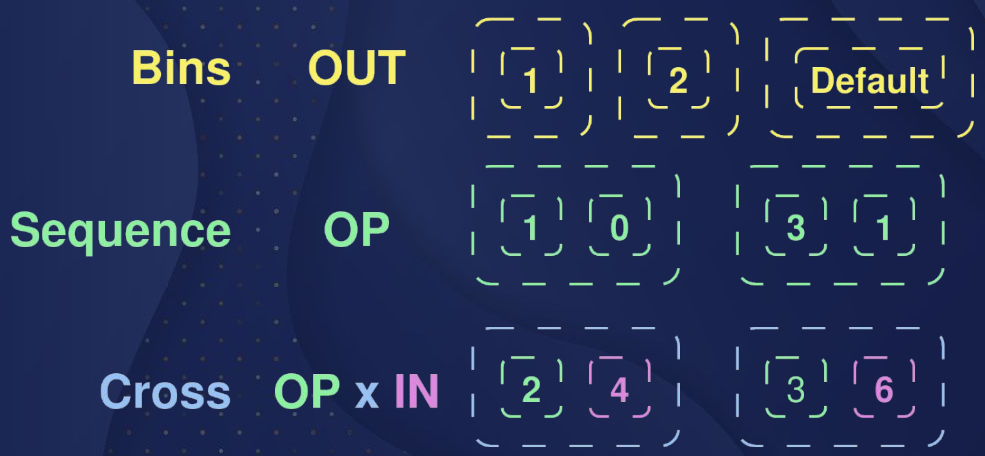
- Engineers are a bit sceptical towards LLMs.
- Coverage is a non-critical part of the testbench.

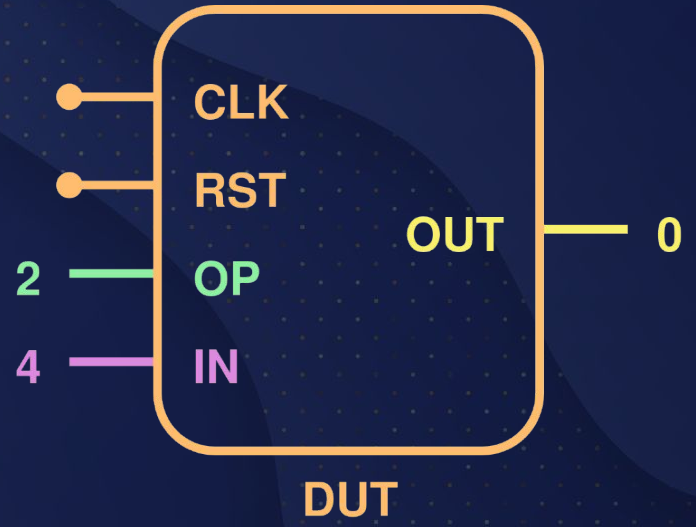
Functional Coverage illustration



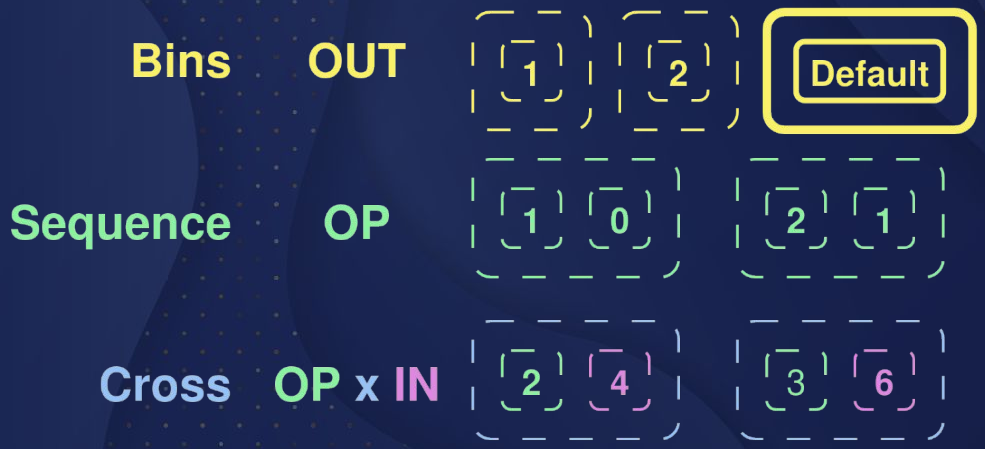


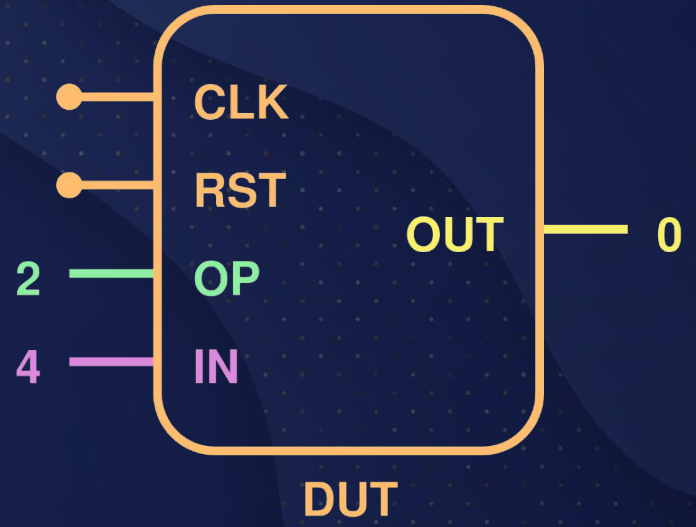
Functional Coverage illustration



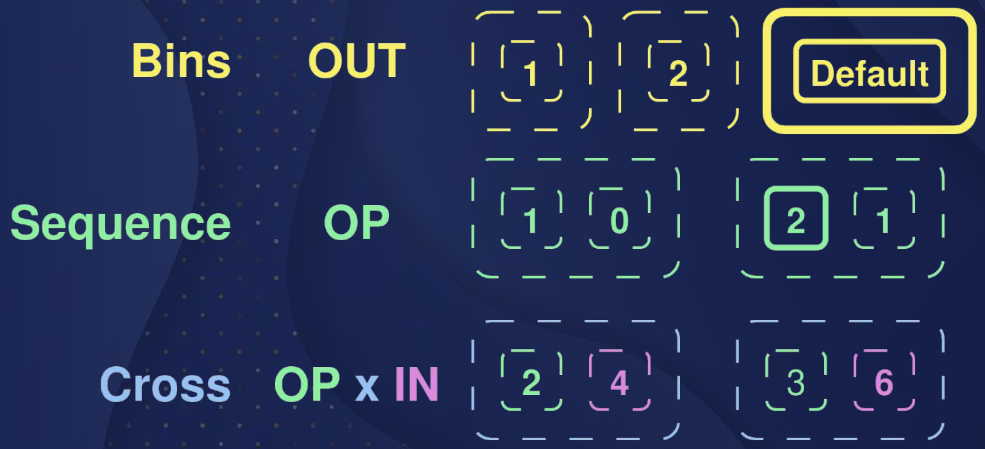


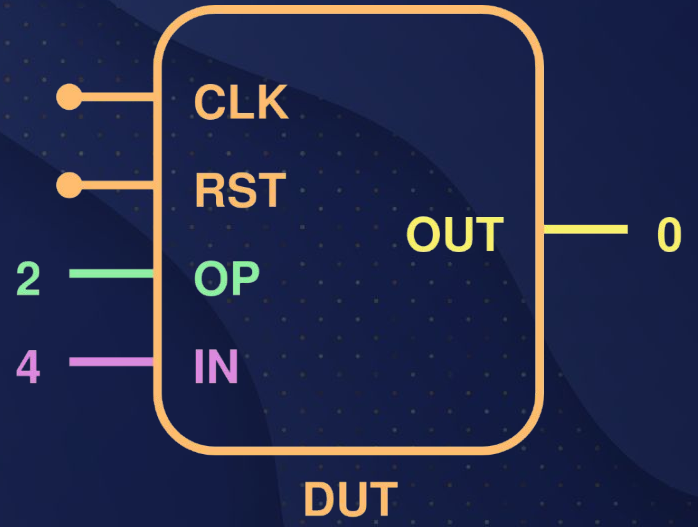
Functional Coverage illustration



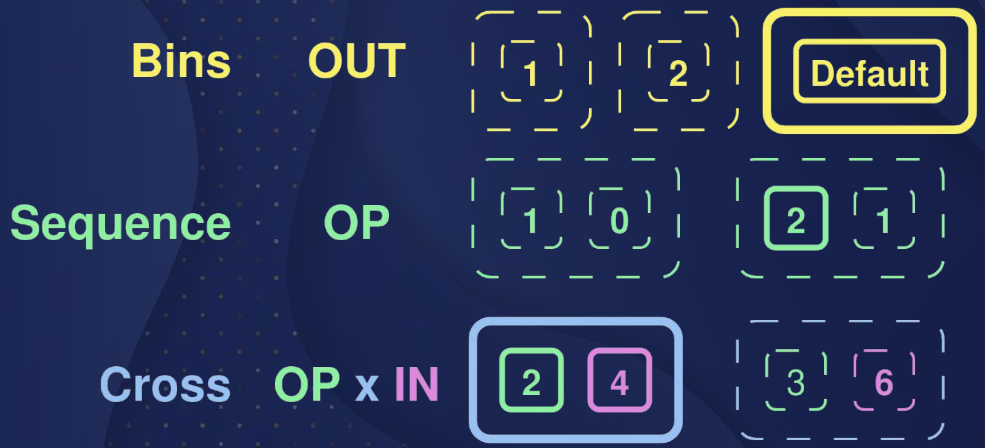


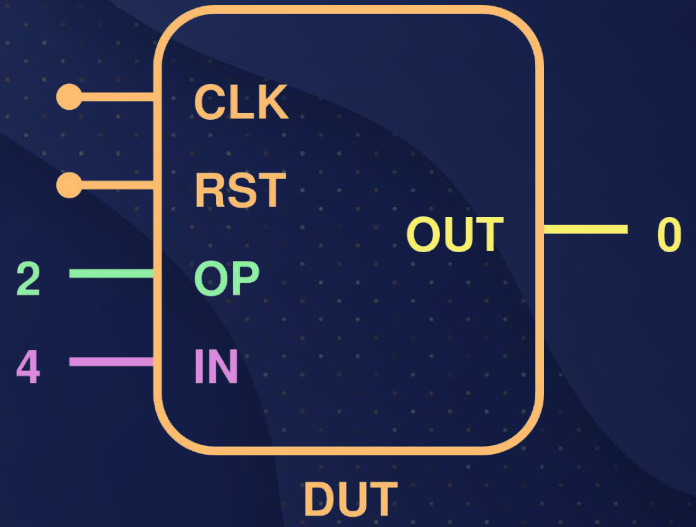
Functional Coverage illustration





Functional Coverage illustration





Functional Coverage and LLMs

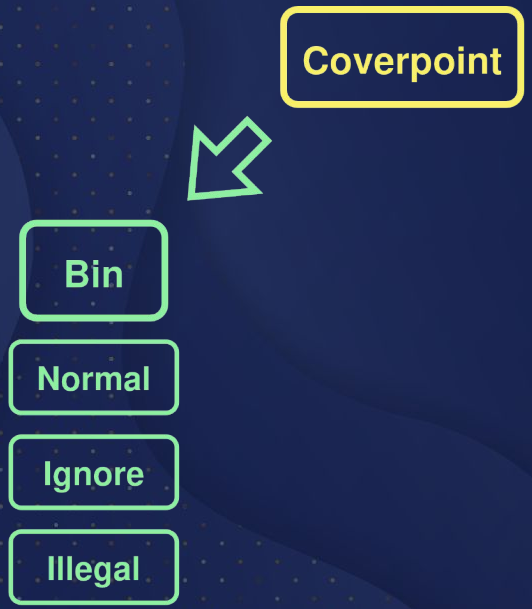
- Initial attempts generated code in SystemVerilog.
- Smaller LLMs struggled to generate syntactically correct code.
- How then evaluate LLMs knowledge about functional coverage?

Our Functional Coverage implementation

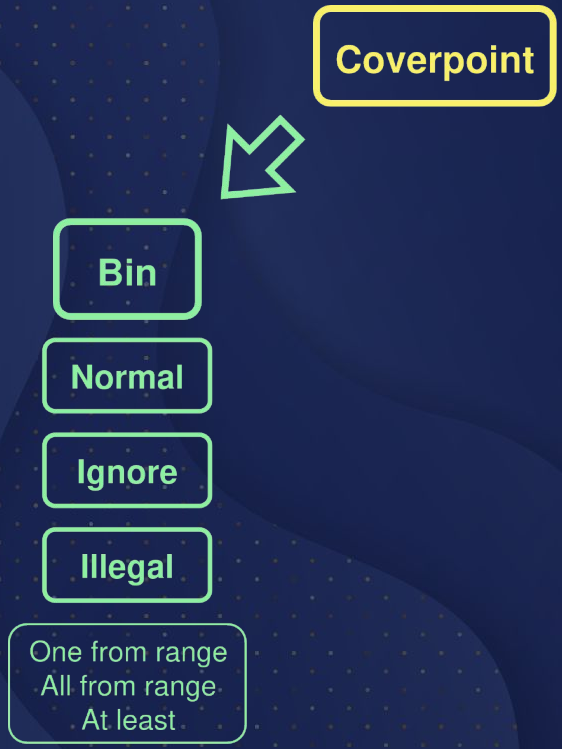
- Python is better understood by LLMs than SystemVerilog.
- Problem: no native support in CoCoTB.
- Available 3rd party package mimicking SystemVerilog.

Functional Coverage API

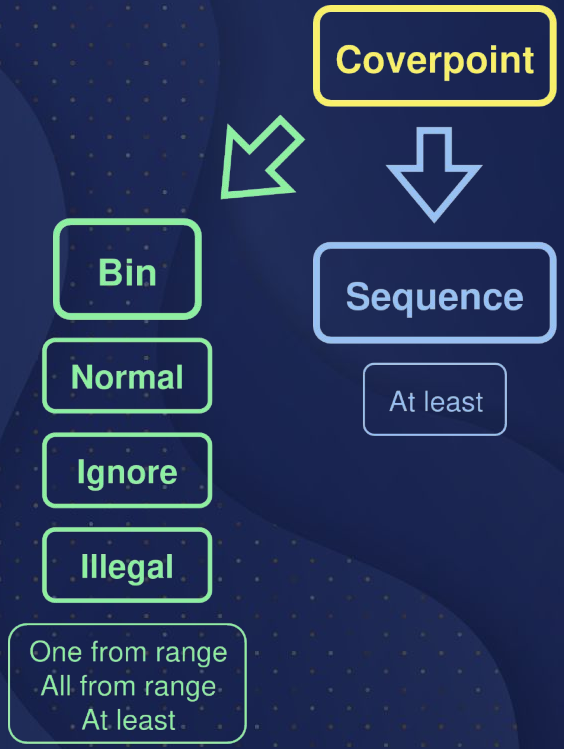
Functional Coverage API



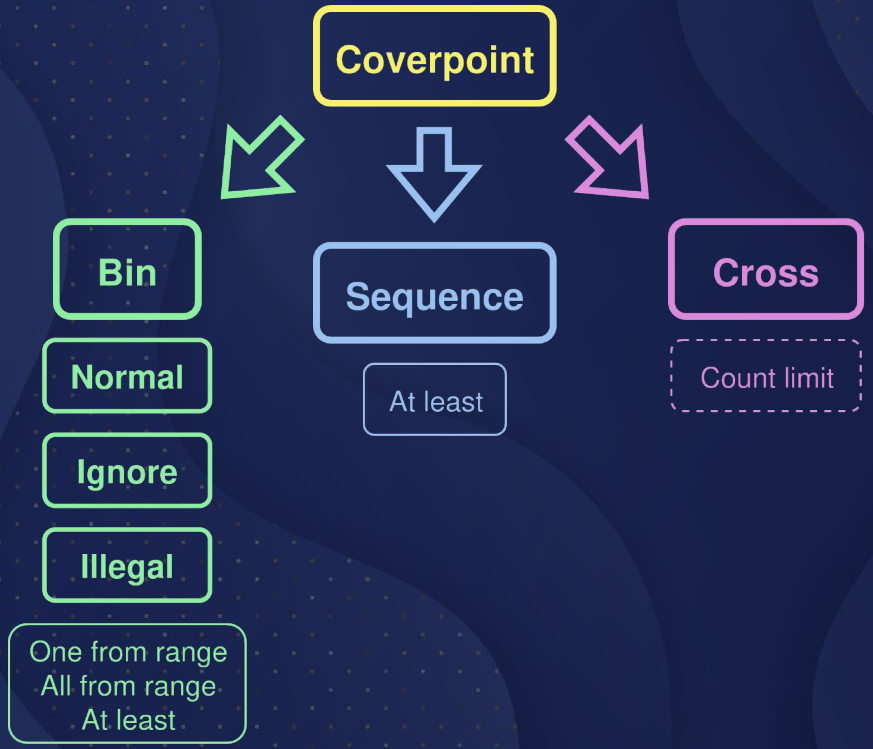
Functional Coverage API



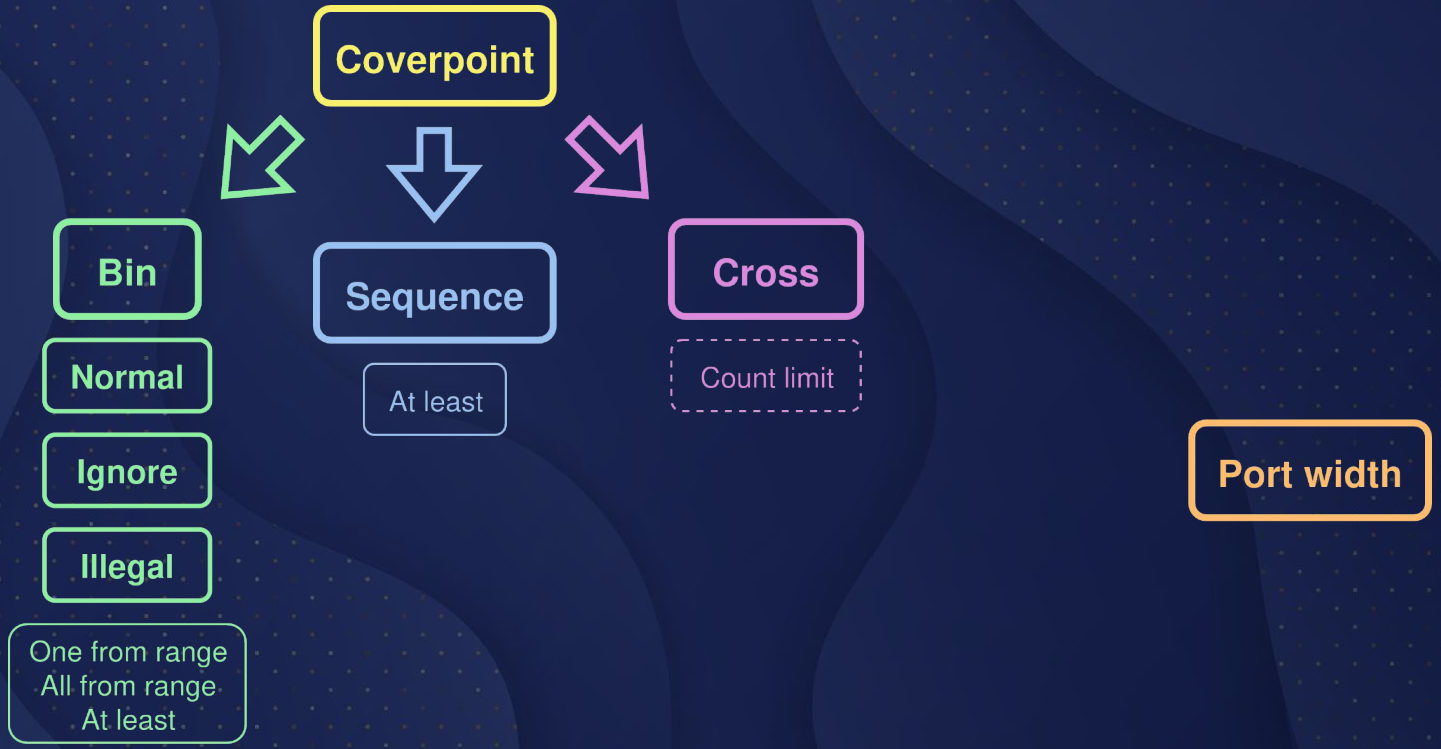
Functional Coverage API



Functional Coverage API



Functional Coverage API



What was tested?

- Top 3 most popular open weight models from Ollama:
  - Deepseek-r1
  - Gemma 3
  - Qwen3
- Various model sizes up to 14 billion parameters.

How it was tested?

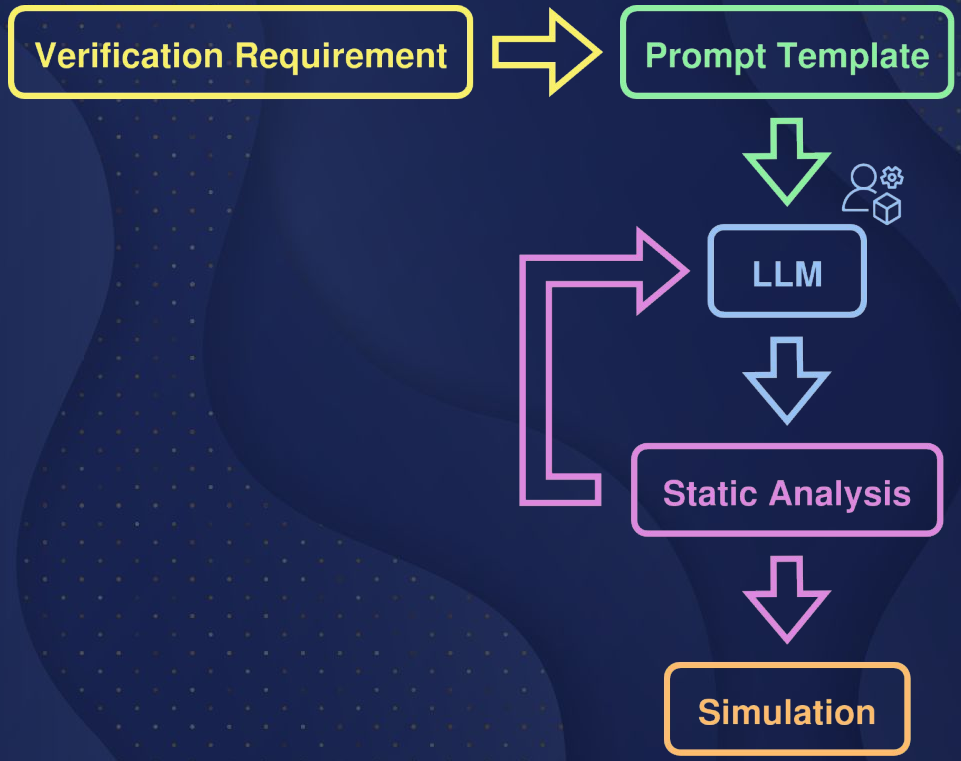
- Initial experiments used natural language specification as an input. (Not successful)
- Shift to natural language verification requirements. (way to go)

How it was tested?

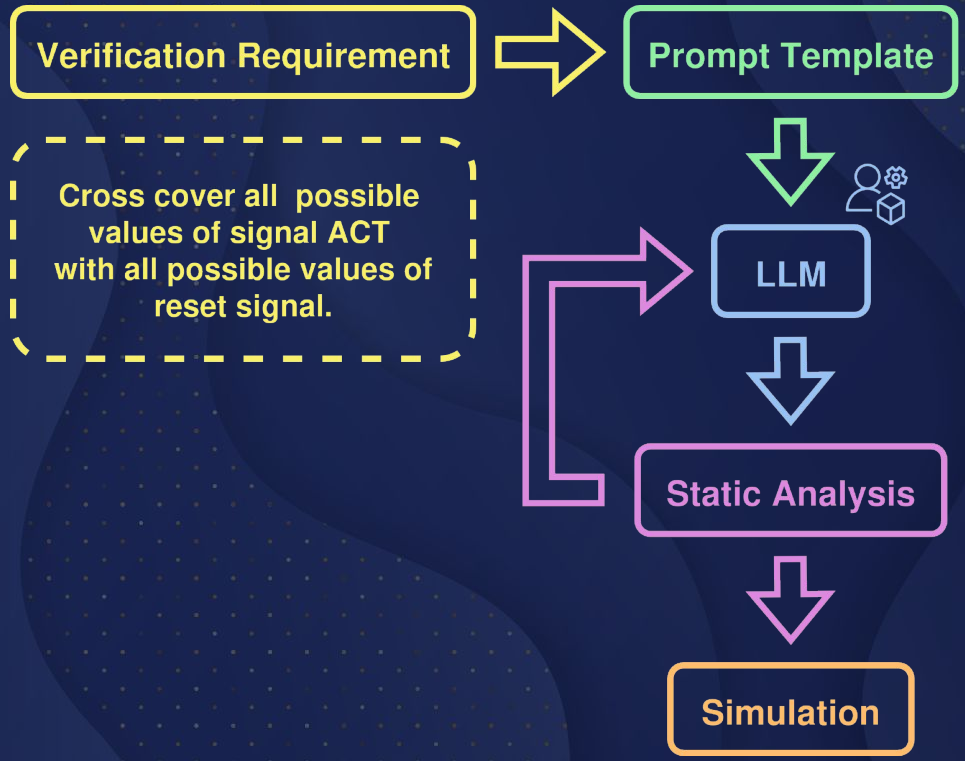
- Initial experiments used natural language specification as an input. (Not successful)
- Shift to natural language verification requirements. (way to go)

- A verification expert provided:
  - 16 verification requirements based on original specification (ALU).
  - Desired functional coverage code.
- Each model with distinct size had 5 attempts that were aggregated.

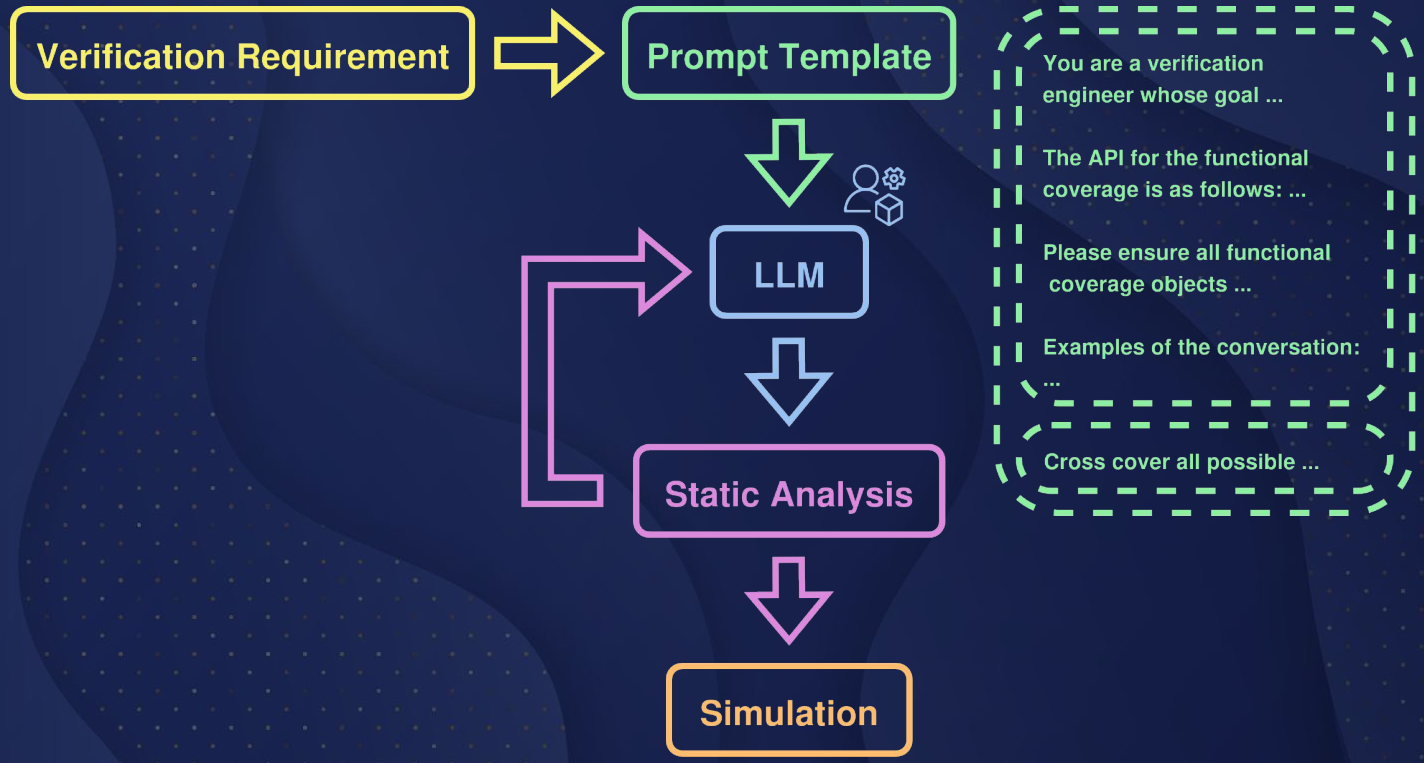
Generation of Functional Coverage



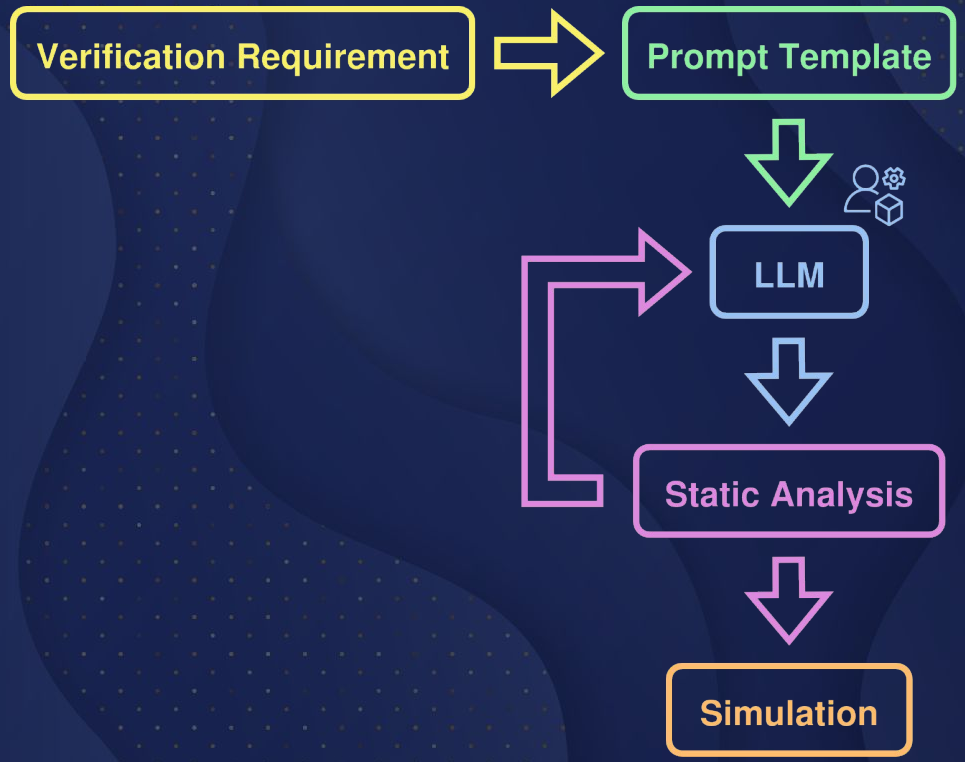
Generation of Functional Coverage



Generation of Functional Coverage

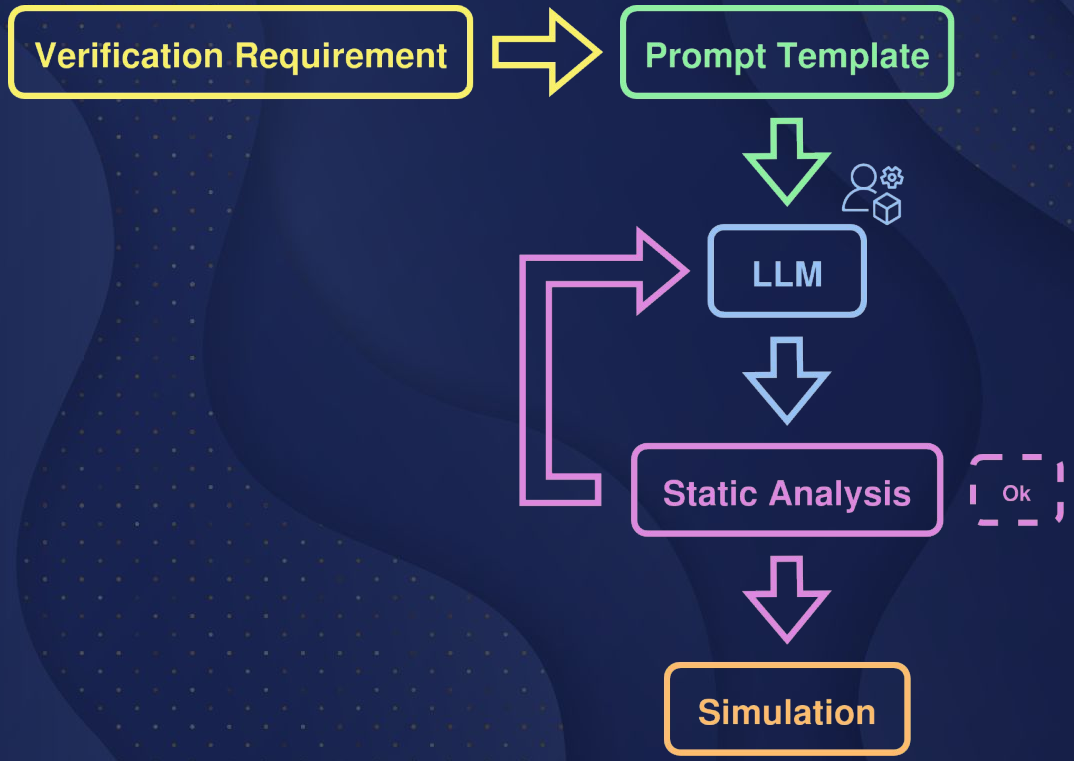


Generation of Functional Coverage

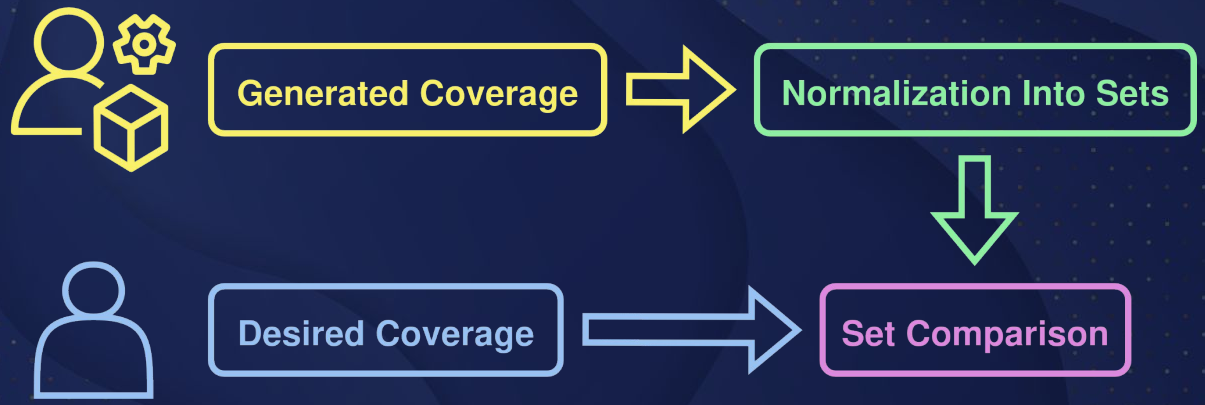



```
...

from coverage import *

act = coverpoint('ACT')
max_act = 2 ** port_width('ACT') - 1
act_bins = normal_bins(act, (0, max_act))

rst = coverpoint('RST')
max_rst = 2 ** port_width('RST') - 1
rst_bins = normal_bins(rst, (0, max_rst))

act_rst_cross = cross([act_bins, rst_bins])
```

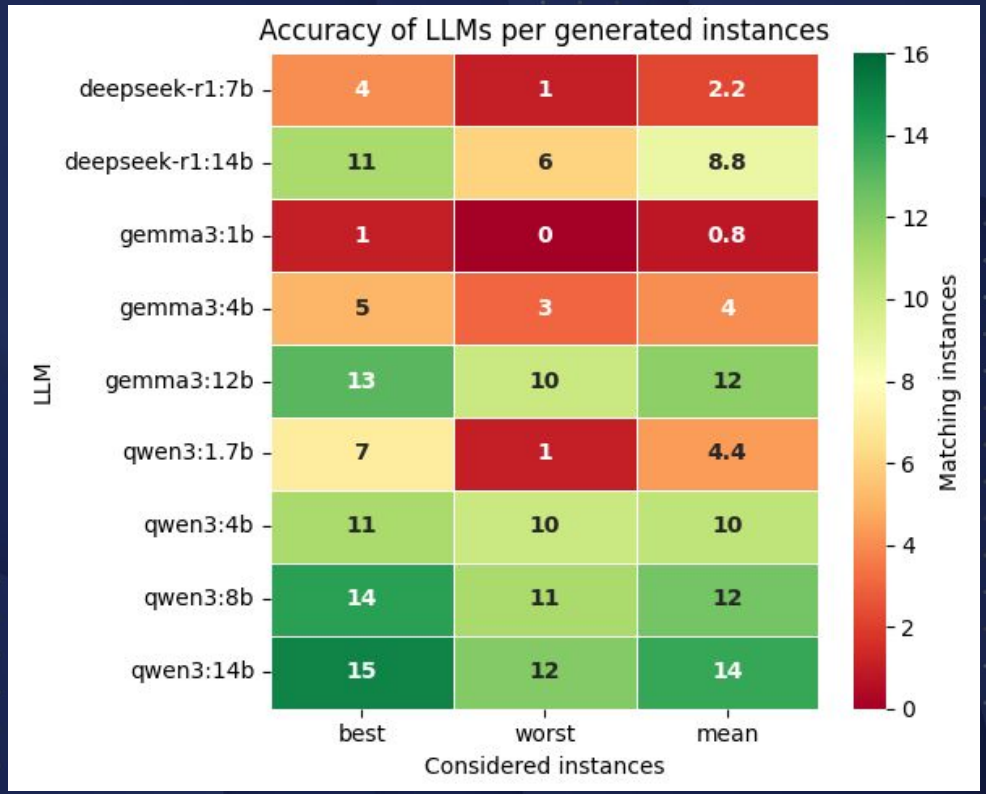
Generation of Functional Coverage



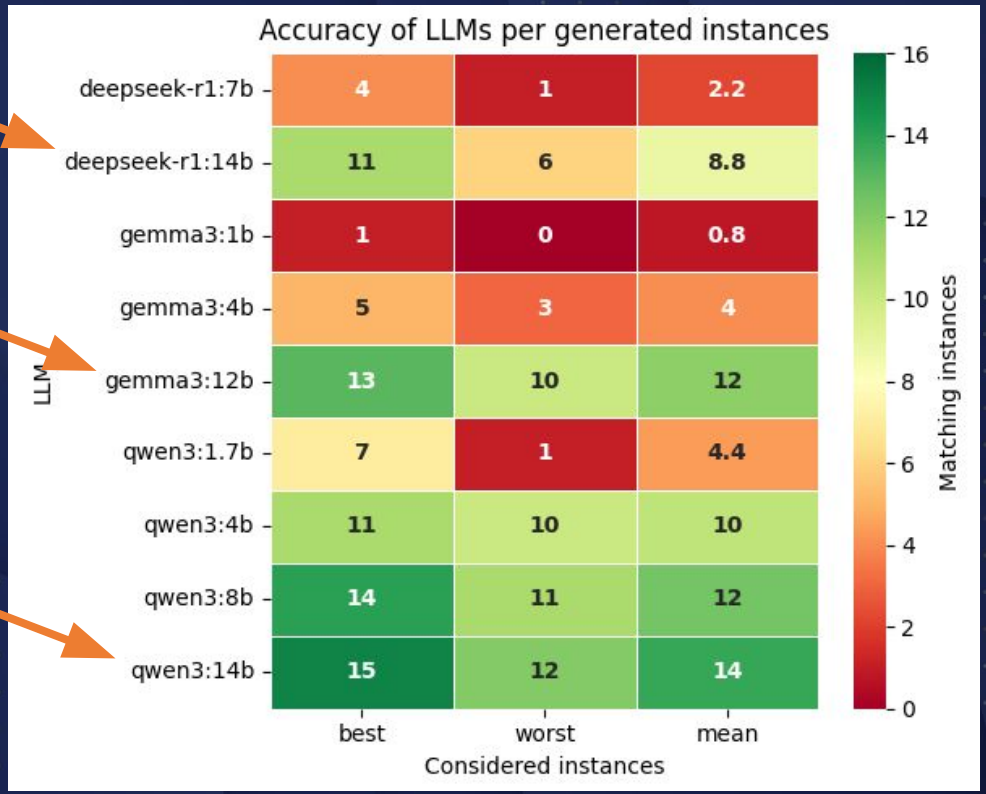
Comparison of Functional Coverage



Results

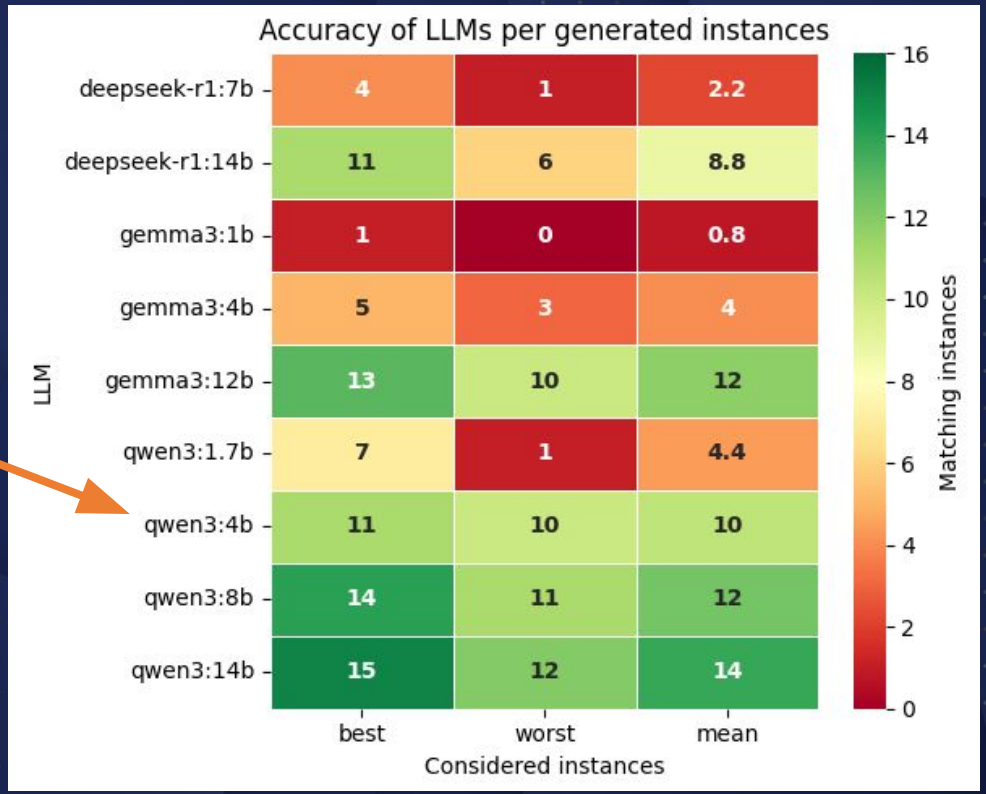

Accuracy of LLMs per generated instances
deepseek-r1:7b
deepseek-r1:14b
gemma3:1b
gemma3:4b
gemma3:12b
qwen3:1.7b
qwen3:4b
qwen3:8b
qwen3:14b
LLM
4
1
2.2
11
6
8.8
1
0
0.8
5
3
4
13
10
12
7
1
4.4
11
10
10
14
11
12
15
12
14
best
worst
mean
Considered instances
Matching instances
16
14
12
10
8
6
4
2
0

Results

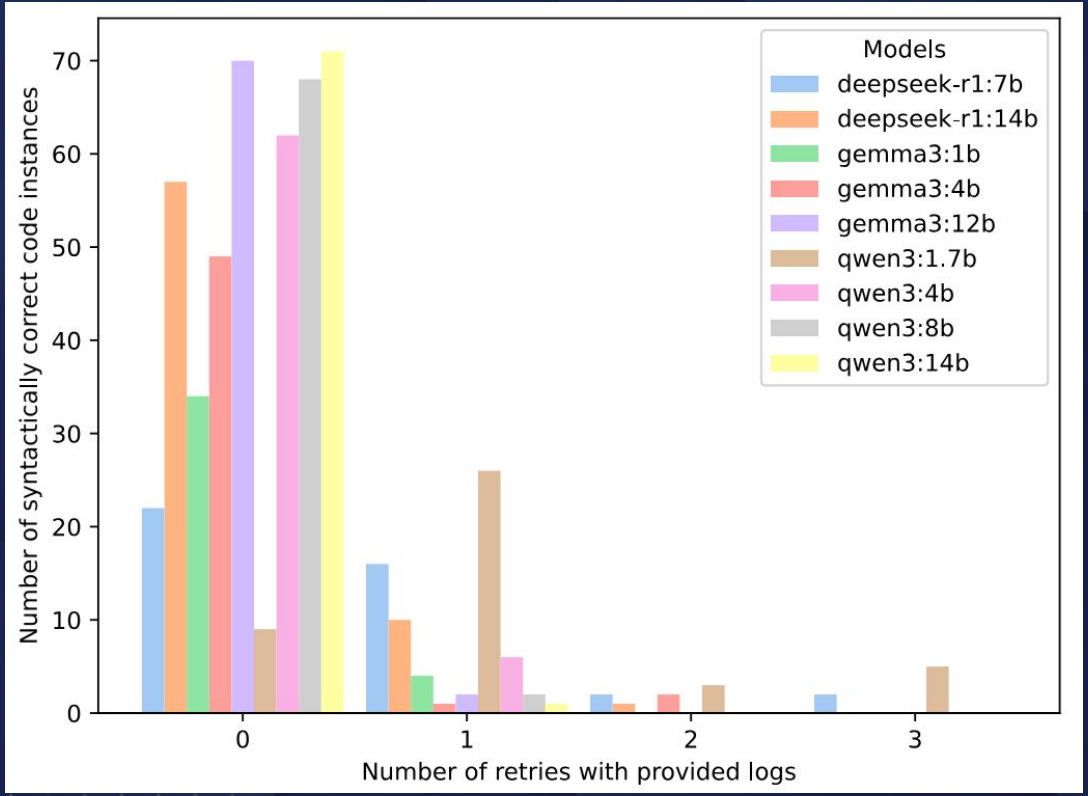

Accuracy of LLMs per generated instances
deepseek-r1:7b
deepseek-r1:14b
gemma3:1b
gemma3:4b
gemma3:12b
qwen3:1.7b
qwen3:4b
qwen3:8b
qwen3:14b
LLM
4
1
2.2
11
6
8.8
1
0
0.8
5
3
4
13
10
12
7
1
4.4
11
10
10
14
11
12
15
12
14
best
worst
mean
Considered instances
Matching instances
16
14
12
10
8
6
4
2
0

Results

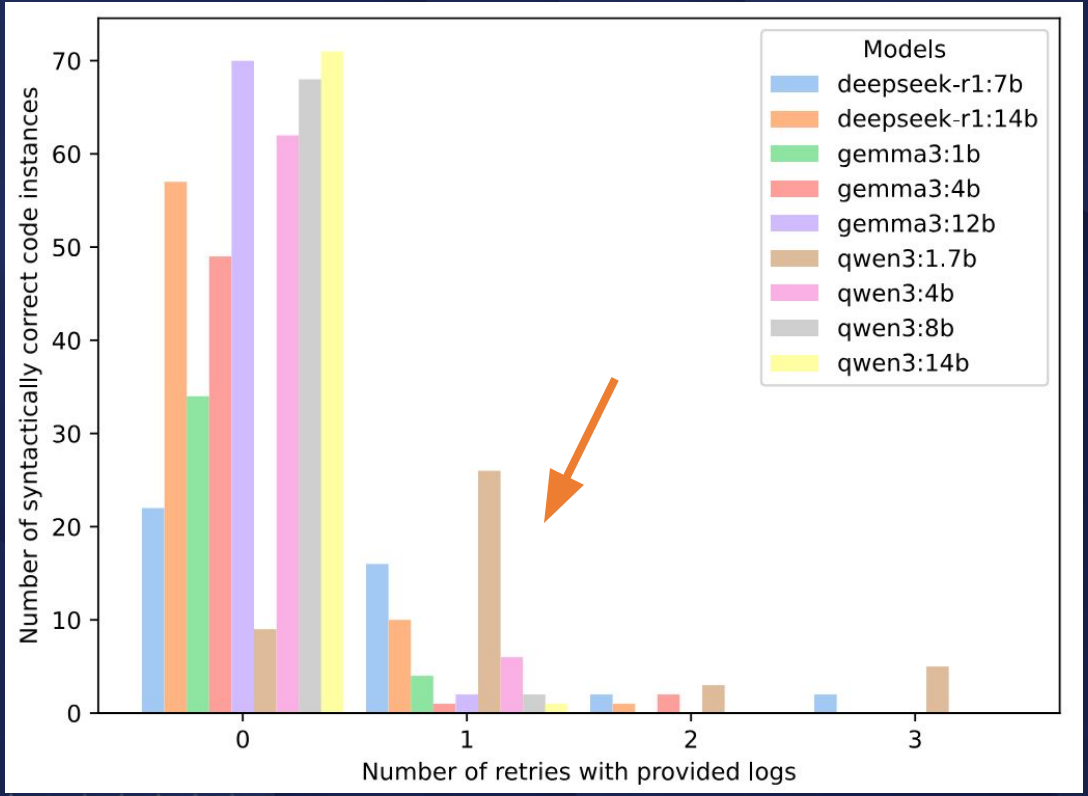
2.5GB VRAM



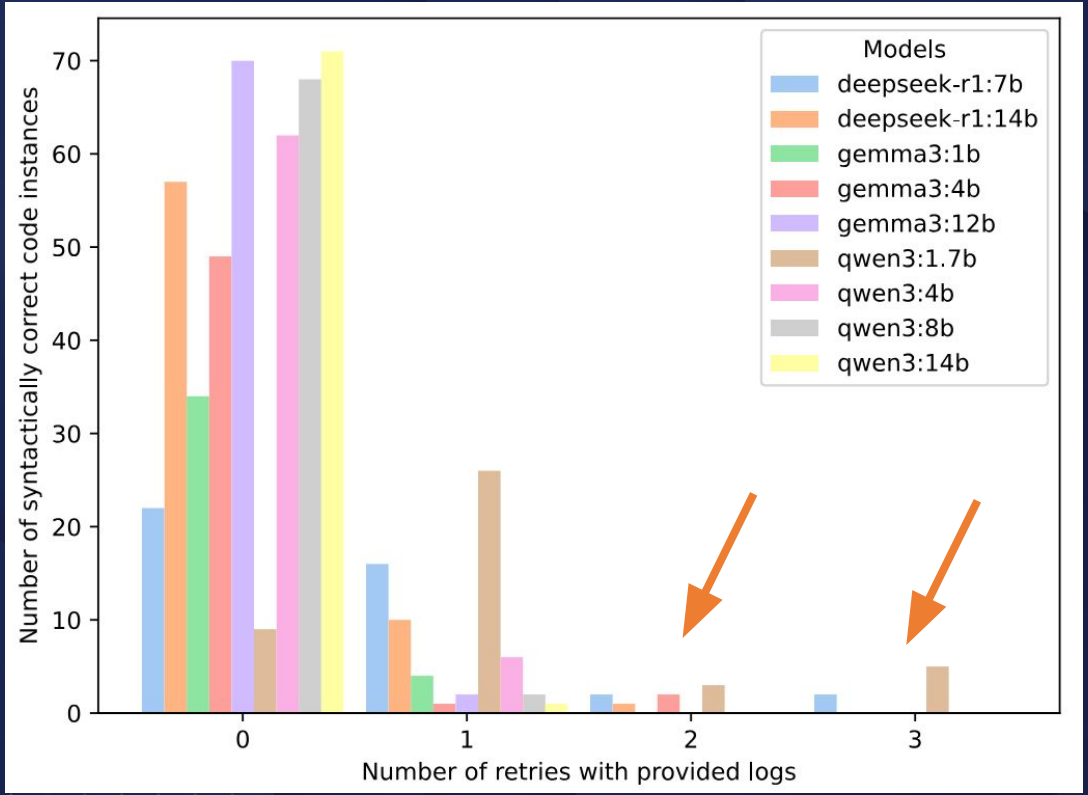
Results


Models
deepseek-r1:7b
deepseek-r1:14b
gemma3:1b
gemma3:4b
gemma3:12b
qwen3:1.7b
qwen3:4b
qwen3:8b
qwen3:14b
Number of syntactically correct code instances
0
10
20
30
40
50
60
70
0
1
2
3
Number of retries with provided logs

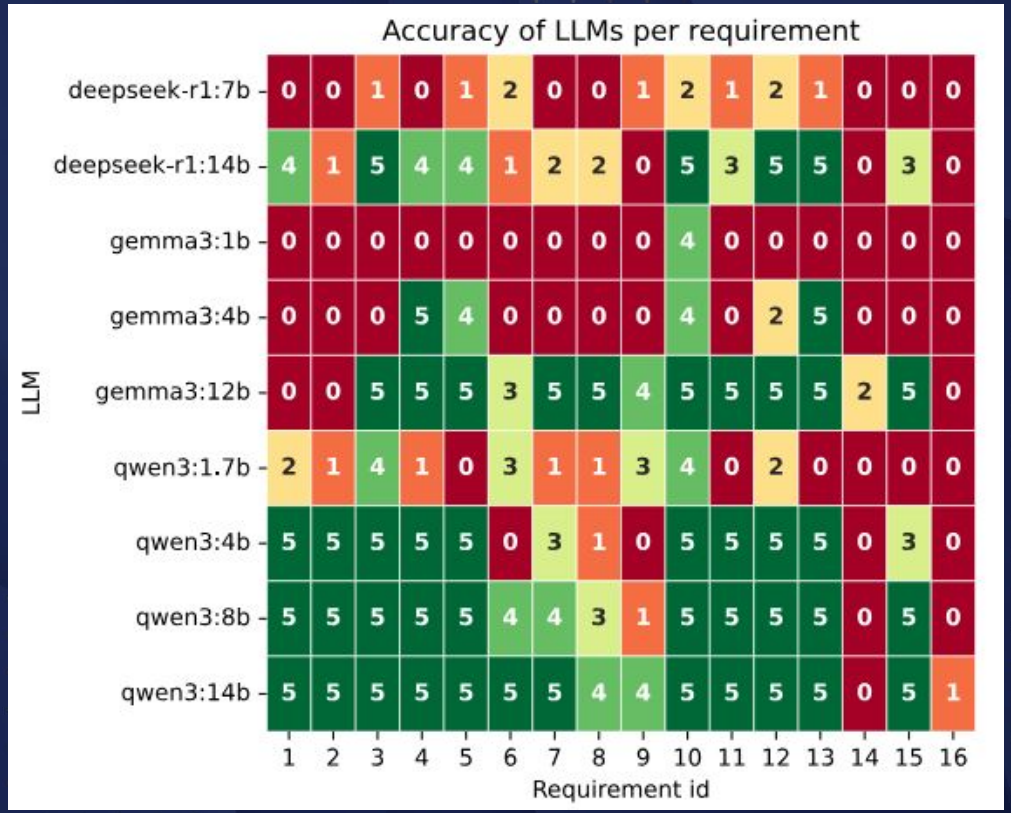
Results


Models
deepseek-r1:7b
deepseek-r1:14b
gemma3:1b
gemma3:4b
gemma3:12b
qwen3:1.7b
qwen3:4b
qwen3:8b
qwen3:14b
Number of syntactically correct code instances
0
10
20
30
40
50
60
70
0
1
2
3
Number of retries with provided logs

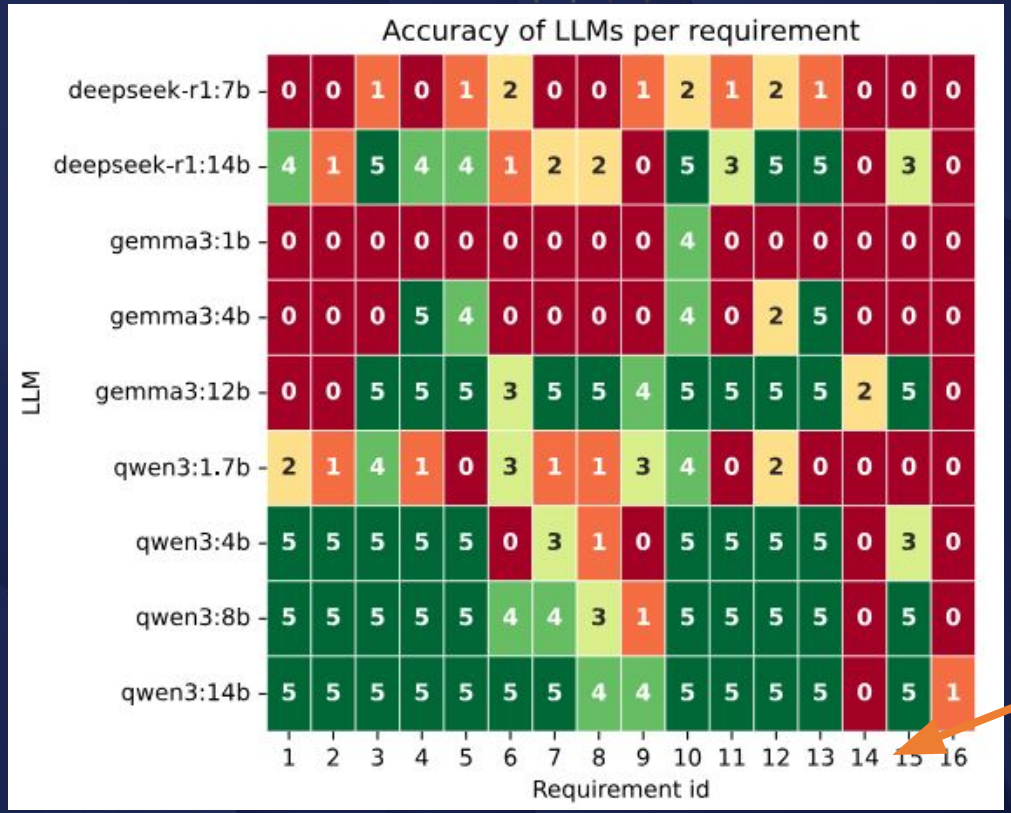
Results


Models
deepseek-r1:7b
deepseek-r1:14b
gemma3:1b
gemma3:4b
gemma3:12b
qwen3:1.7b
qwen3:4b
qwen3:8b
qwen3:14b
Number of syntactically correct code instances
0
10
20
30
40
50
60
70
0
1
2
3
Number of retries with provided logs

Results

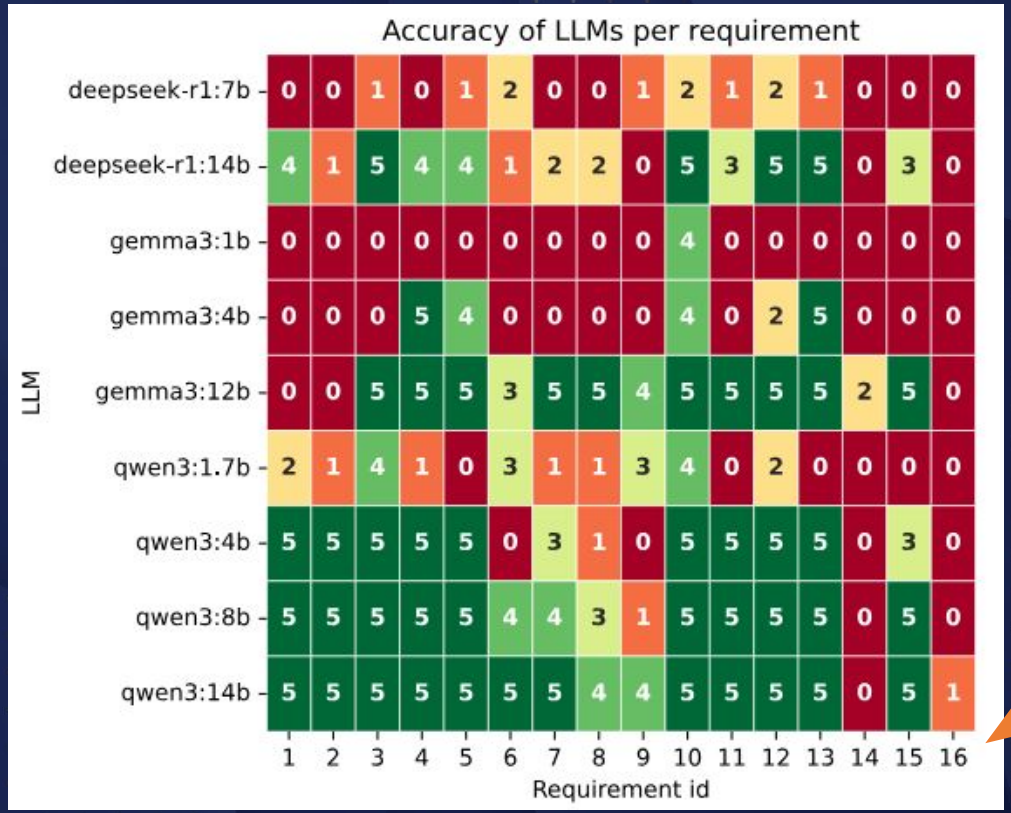

Accuracy of LLMs per requirement
deepseek-r1:7b 0 0 1 0 1 2 0 0 1 2 1 2 1 0 0 0
deepseek-r1:14b 4 1 5 4 4 1 2 2 0 5 3 5 5 0 3 0
gemma3:1b 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0
gemma3:4b 0 0 0 5 4 0 0 0 0 4 0 2 5 0 0 0
gemma3:12b 0 0 5 5 5 3 5 5 4 5 5 5 5 2 5 0
qwen3:1.7b 2 1 4 1 0 3 1 1 3 4 0 2 0 0 0 0
qwen3:4b 5 5 5 5 5 0 3 1 0 5 5 5 5 0 3 0
qwen3:8b 5 5 5 5 5 4 4 3 1 5 5 5 5 0 5 0
qwen3:14b 5 5 5 5 5 5 5 4 4 5 5 5 5 0 5 1
LLM
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Requirement id

Results



Generated too many bins

Results



Cross coverage was too large

What now?

- Future plans:
  - Enhance our open source dataset with more samples.
  - Evaluate generation of the SystemVerilog functional coverage.
  - Pre-train and finetune LLMs on synthetic data.
- Looking for a PhD research stay - let’s discuss!
- Try the code from github.com/Northeus/coge
- Contact: jan.labuda@mail.muni.cz