

AI for Chip Design

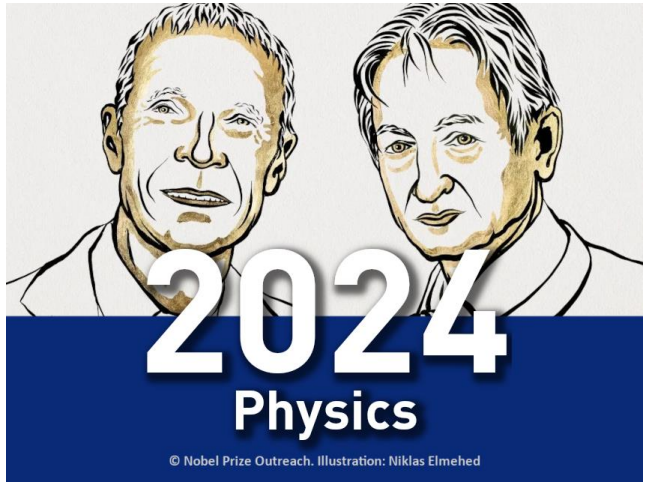
Haoxing (Mark) Ren, Director of Design Automation Research, NVIDIA

11/20/2024

AI Researchers won Nobel Prizes

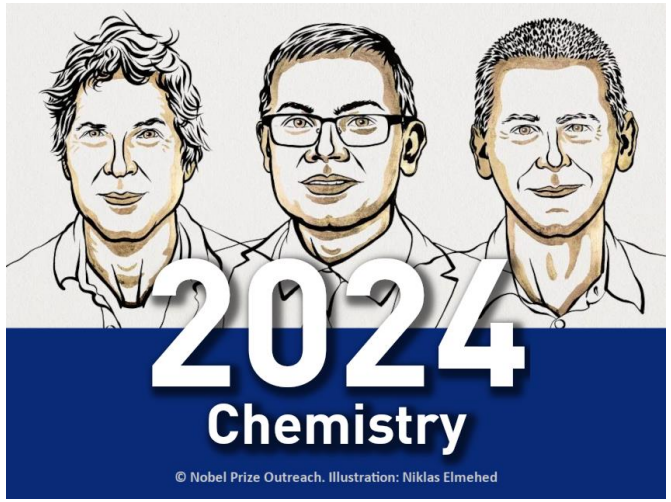
The story is “ Science helps create AI, and AI helps develop Science”

What is our story? “Chip helps build AI, and AI helps design Chip”



Hopfield and Hinton

They trained artificial neural networks using physics



Baker and Hassabis and Jumper

They cracked the code for proteins’ amazing structures

NVIDIA



Agenda

Introduction of AI for Chip Design

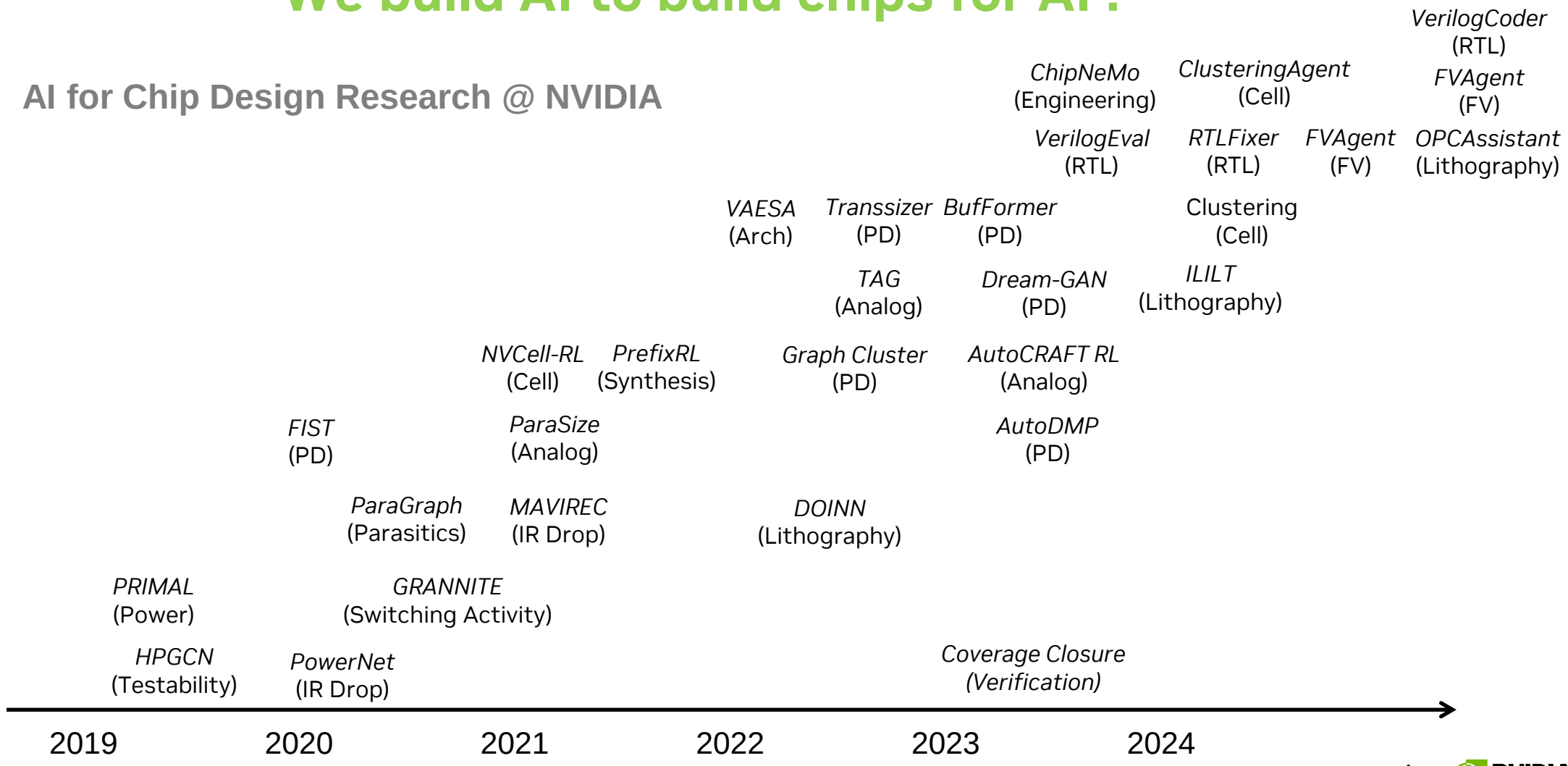
LLM for Verilog Generation

Research Opportunities

NVIDIA

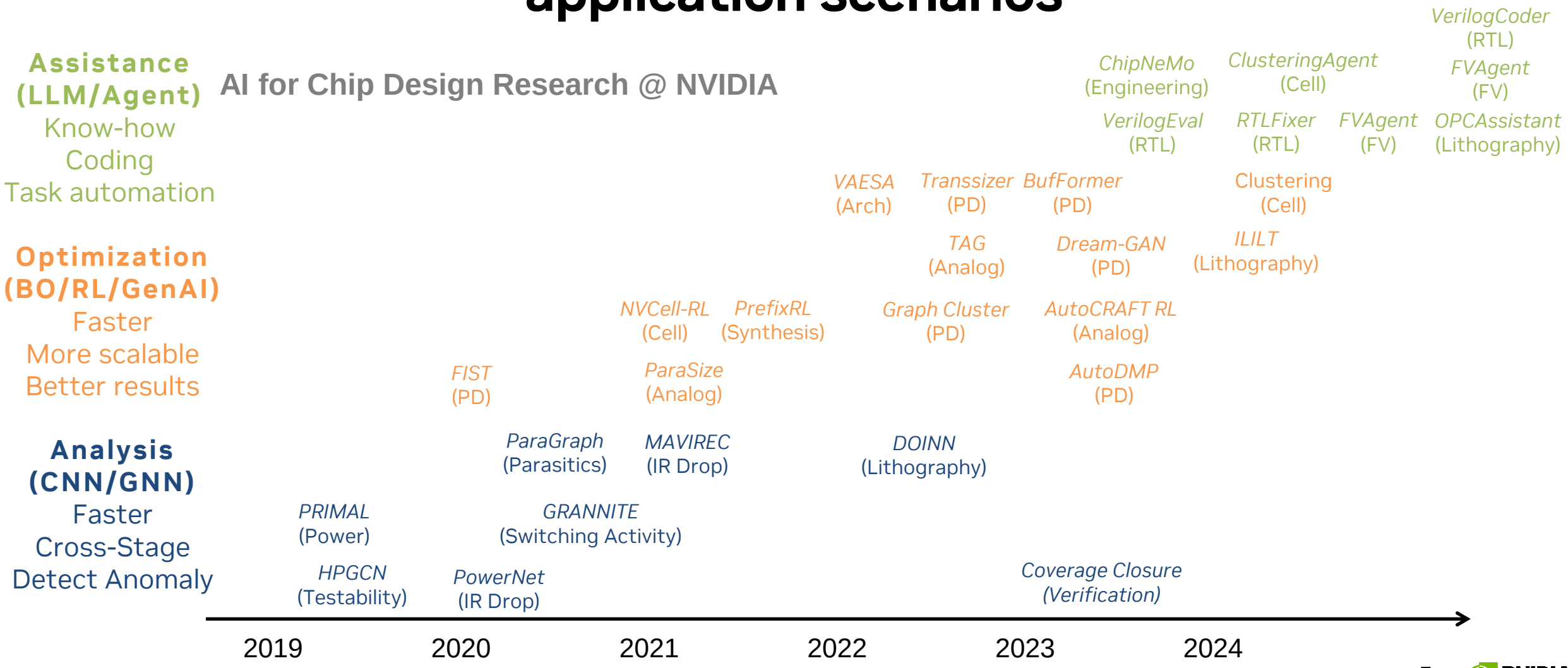
AI works for many chip design problems

We build AI to build chips for AI !



NVIDIA

Analysis, Optimization, and Assistance are the main application scenarios



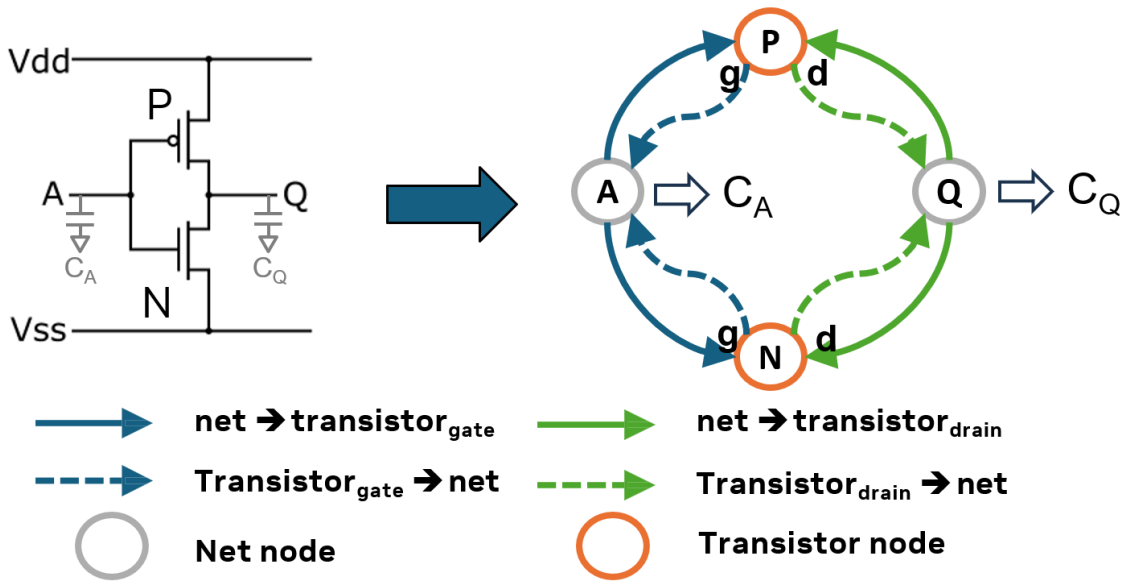
AI predicts cross-stage design metrics

Impact of layout parasitics on schematic design

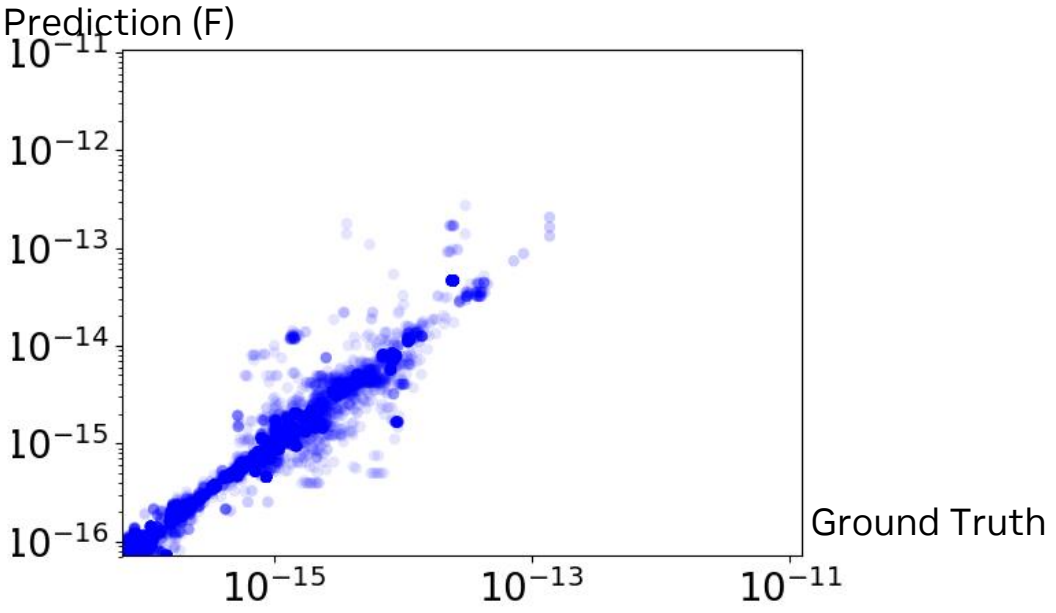
Use AI to predict layout parasitics from schematic given a large dataset

Convert schematic to graph and learn with GNN

Predicted parasitics reduced simulation error to <10%



Circuit Schematics to Heterogenous Graph Conversion



MAE=0.852fF MAPE=15%

H. Ren et al, ParaGraph: Layout Parasitics and Device Parameter Prediction using Graph Neural Networks

NVIDIA

AI optimizes tools parameters

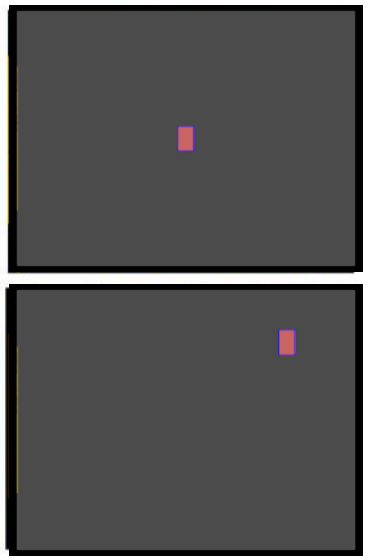
Macro placement quality is very important for physical design

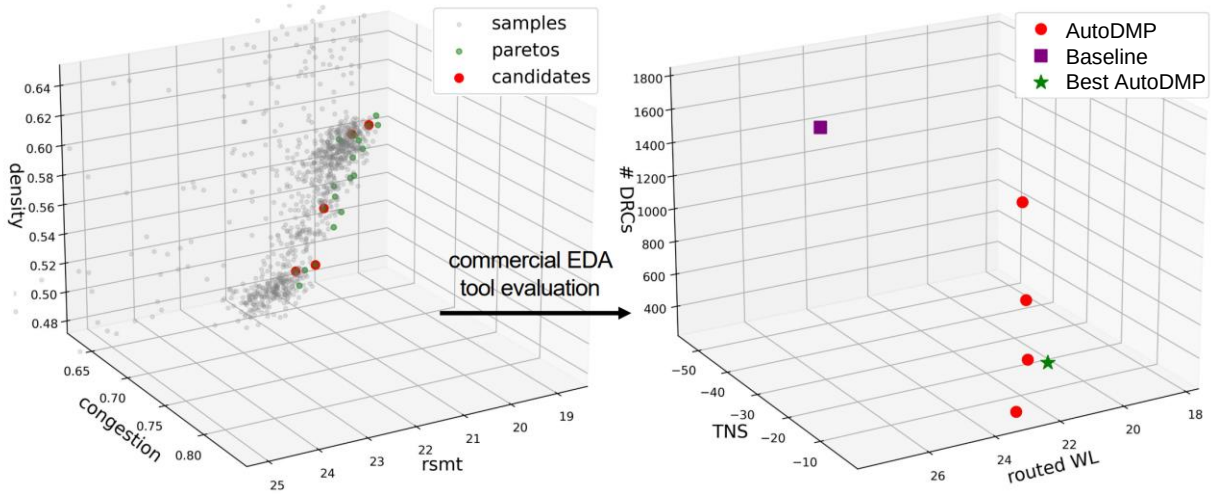
Placement tools parameters have a huge impact on macro placement

Multi-objective Bayesian Optimization (sample and model) find best parameter set

Find better macro placement with open-source GPU accelerated placement tools

Parameter	Search Range
*horiz. initial position *vert. initial position	[0.2, 0.8] (%) [0.2, 0.8] (%)
*horiz. macro halo *vert. macro halo	technology dep. technology dep.
target density d_{target} density weight	$[a_{\text{util}} - 0.2, a_{\text{util}}]$ (%) $[1e^{-6}, 1.0]$
smooth HPWL model smooth HPWL initial γ_0	{LSE, WA} [0.10, 0.50]
GD initial LR lr_0 GD LR decay GD optimizer	$[1e^{-4}, 1e^{-2}]$ [0.99, 1.0] [Adam, Nesterov]
# horiz. global bins # vert. global bins	{256, 512, 1024, 2048} {256, 512, 1024, 2048}
λ update lower coeff. L λ update upper coeff. U λ update Δ HPWL_{REF}	[0.90, 0.99] [1.01, 1.15] $[1.5e^5, 5.5e^5]$





A. Agnesina et al, AutoDMP: Automated DREAMPlace-based Macro Placement

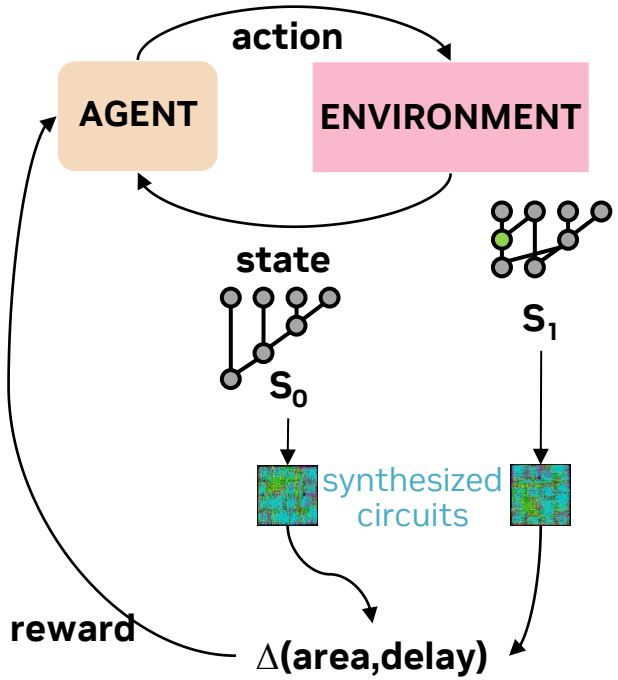
NVIDIA

AI optimizes datapath structure

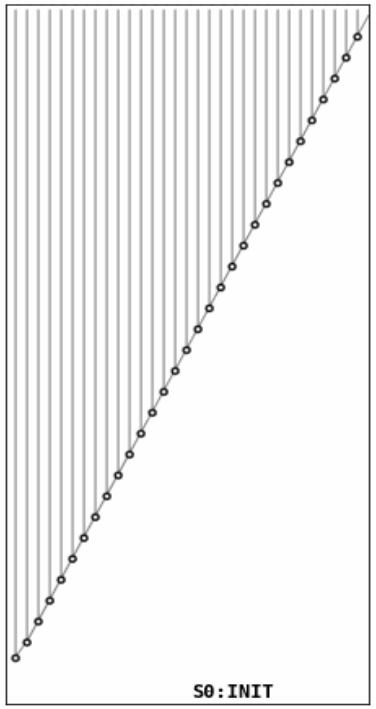
Datapath synthesis important for GPU

Optimize prefix adder structure with Reinforcement Learning for better delay and area tradeoff

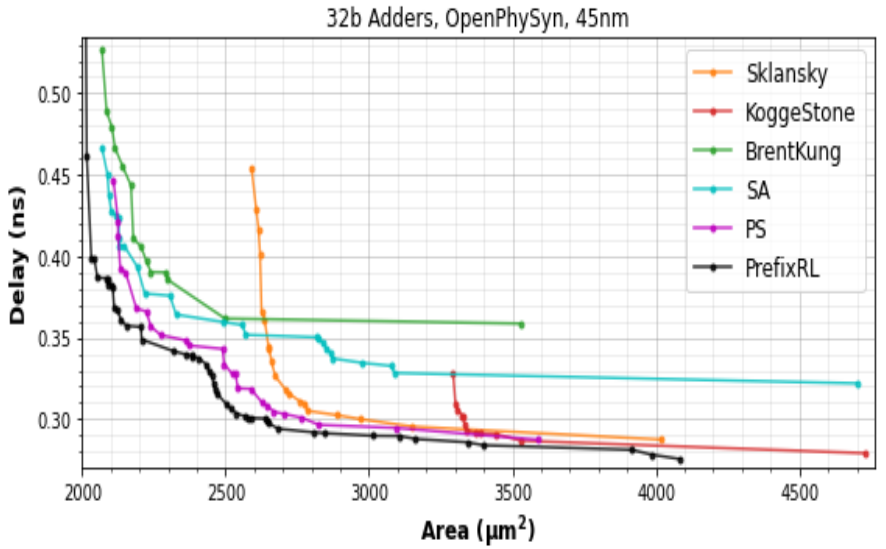
Reward computed by open-source EDA tools



Deep Q learning
Circuit synthesis in-the-loop



Action Space
Add/Delete prefix graph nodes



PrefixRL achieves better results than well known adder architectures

R. Roy et al, PrefixRL: Optimization of parallel prefix circuits using deep reinforcement learning

NVIDIA

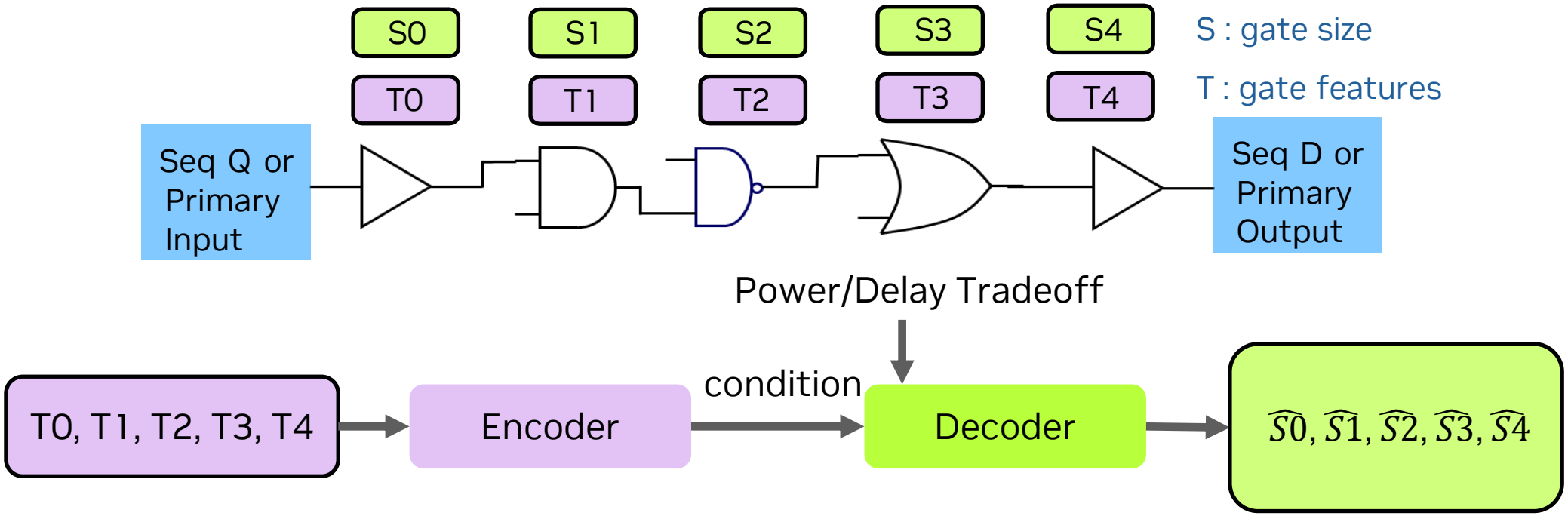
AI Generates Optimal Gate Sizes

Timing/power optimization such as gate sizing affects scalability of PD tools

Model a path of gates as a sequence, generate optimized gate sizes using Transformer

Trained on tool optimized netlist dataset

100X – 1000X speedup compared to traditional optimization with similar PPA



S. Nath et al, Transizer: A Novel Transformer-Based Fast Gate Size

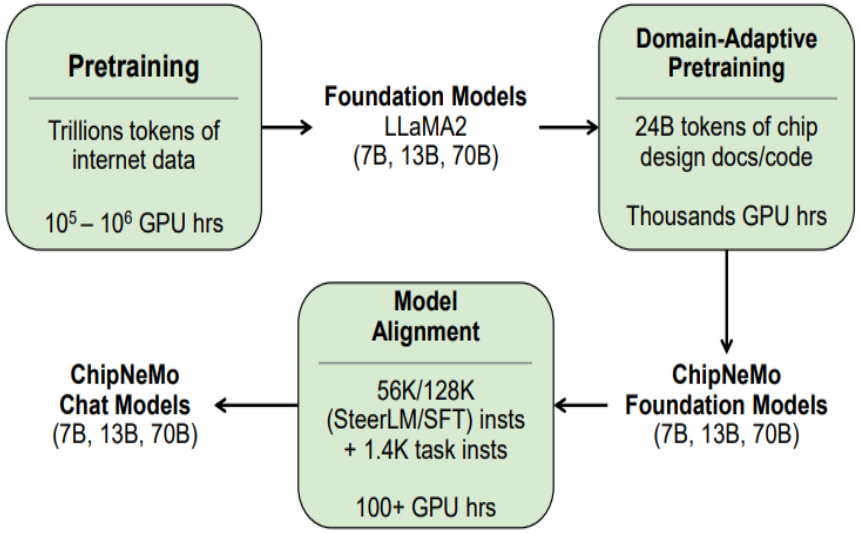
NVIDIA

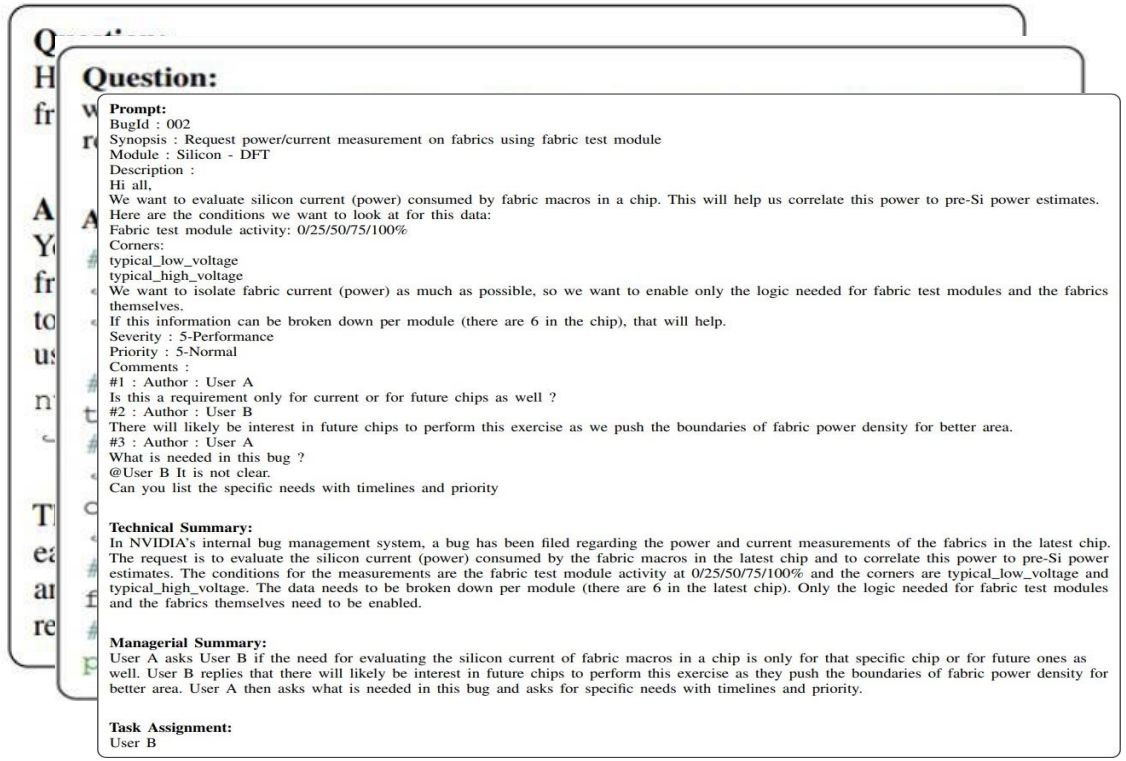
AI assists engineers in QnA, coding and analysis tasks

LLM is good at question answering, coding, extraction, rewriting, summarization, classification, reasoning, …

- Answer questions about designs, infrastructures, tools, flows, HW domains, etc.
- Generate scripts for specific tasks (VLSI)
- Summarize bug reports, predict assignment

Domain-adapted model have better performance





Question:

Prompt:
BugId : 002
Synopsis : Request power/current measurement on fabrics using fabric test module
Module : Silicon - DFT
Description :
Hi all,
We want to evaluate silicon current (power) consumed by fabric macros in a chip. This will help us correlate this power to pre-Si power estimates.
Here are the conditions we want to look at for this data:
Fabric test module activity: 0/25/50/75/100%
Corners:
typical_low_voltage
typical_high_voltage
We want to isolate fabric current (power) as much as possible, so we want to enable only the logic needed for fabric test modules and the fabrics themselves.
If this information can be broken down per module (there are 6 in the chip), that will help.
Severity : 5-Performance
Priority : 5-Normal
Comments :
#1 : Author : User A
Is this a requirement only for current or for future chips as well ?
#2 : Author : User B
There will likely be interest in future chips to perform this exercise as we push the boundaries of fabric power density for better area.
#3 : Author : User A
What is needed in this bug ?
@User B It is not clear.
Can you list the specific needs with timelines and priority

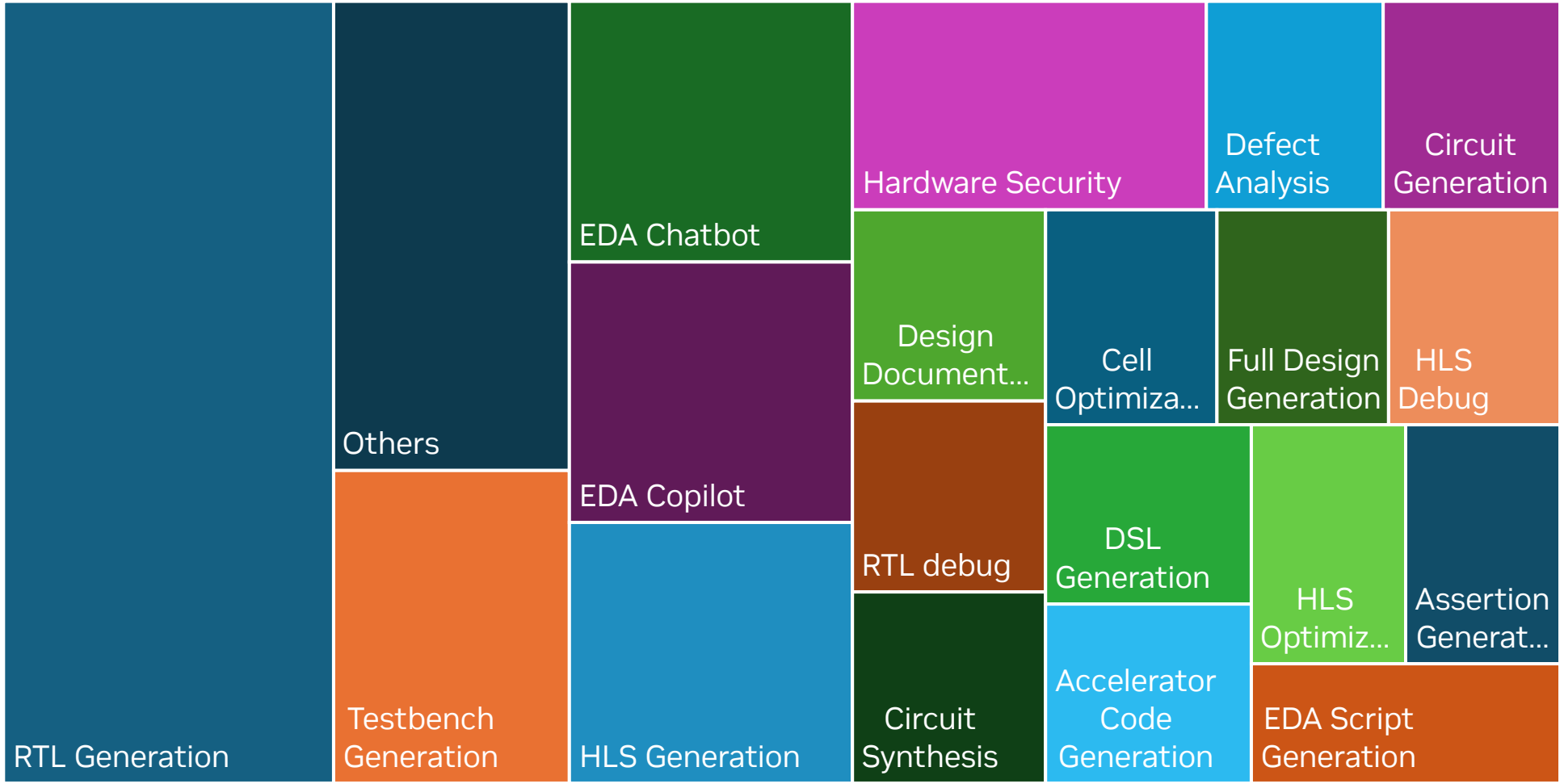
Technical Summary:
In NVIDIA's internal bug management system, a bug has been filed regarding the power and current measurements of the fabrics in the latest chip. The request is to evaluate the silicon current (power) consumed by the fabric macros in the latest chip and to correlate this power to pre-Si power estimates. The conditions for the measurements are the fabric test module activity at 0/25/50/75/100% and the corners are typical_low_voltage and typical_high_voltage. The data needs to be broken down per module (there are 6 in the latest chip). Only the logic needed for fabric test modules and the fabrics themselves need to be enabled.

Managerial Summary:
User A asks User B if the need for evaluating the silicon current of fabric macros in a chip is only for that specific chip or for future ones as well. User B replies that there will likely be interest in future chips to perform this exercise as they push the boundaries of fabric power density for better area. User A then asks what is needed in this bug and asks for specific needs with timelines and priority.

Task Assignment:
User B

M. Liu et al, ChipNeMo: Domain-Adapted LLMs for Chip Design

LLM for Chip Design – Cambrian Explosion



Papers in ISLAD 2024 (LLM-Aided Design) : islad.org



Agenda

Introduction of AI for Chip Design

LLM for Verilog Generation

- **Better model**: fine-tune model with synthetic data
- **Better inference:** Agents

Research Opportunities

LLM Applications for Chip Design



Coding Assistance → Generating design and verification code (RTL, SVA, testbenches, EDA scripts, tools scripts, and configs)

- Generate code from specifications
- Generate code for auxiliary design tasks such as assertions, comments, etc.
- Generate lower-level programs from higher-level descriptions
- Generate scripts for specific tasks (VLSI, Verification)
- Transform code for more efficient implementation

Know-how Assistance → Generating insights, knowledge, and ideas. Answer questions about designs, infrastructures, tools, flows, HW domains, etc.

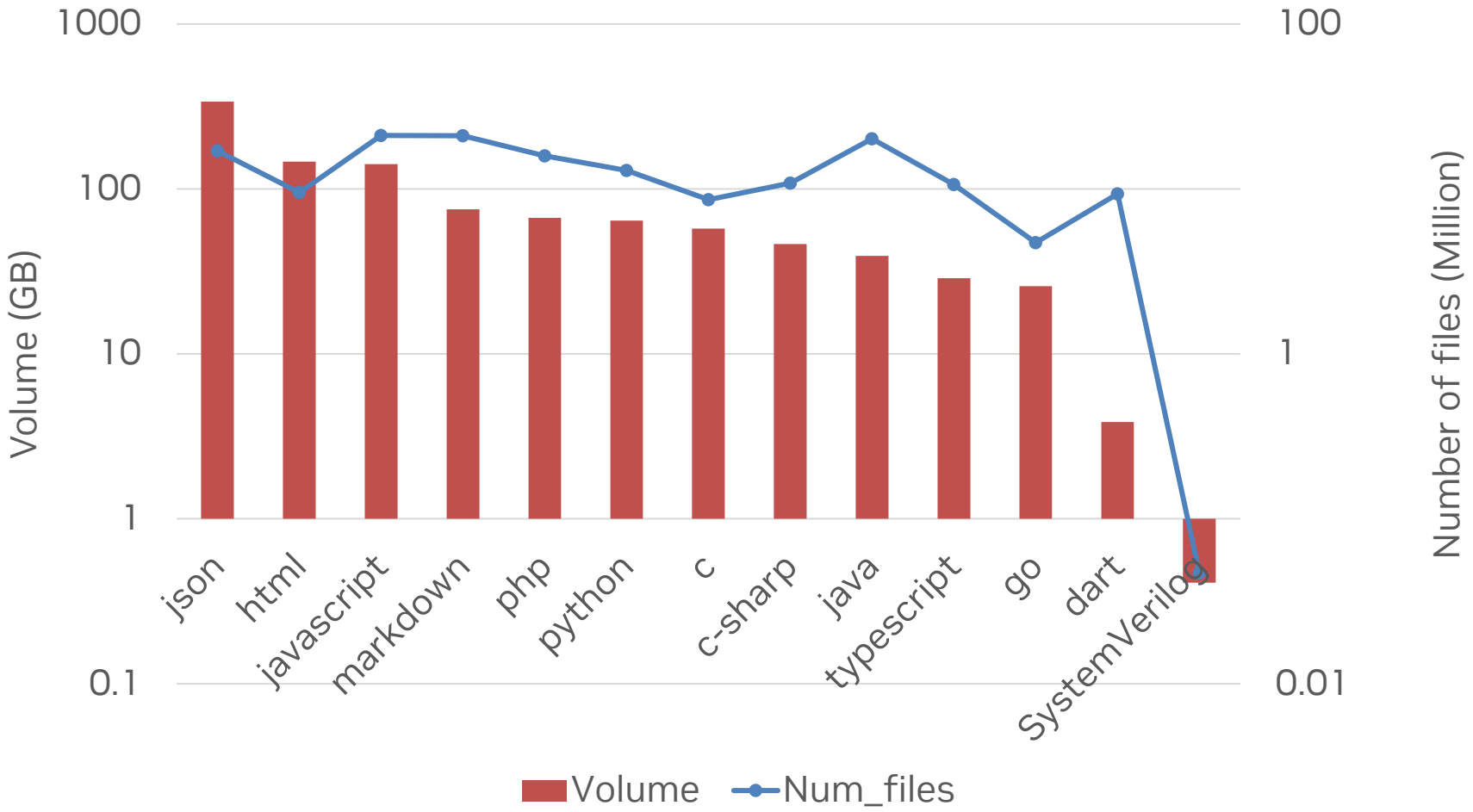
Analysis Assistance → Summarization, report/log analysis, visualization of design and related data, check rule violations, etc.

Debug Assistance → Triage and fix a design problem, e.g. regression failures for logic design and timing failures for physical design.

Optimization Assistance → Optimize for design and verification: coverage closure, recipe optimization, architecture exploration, etc.

NVIDIA

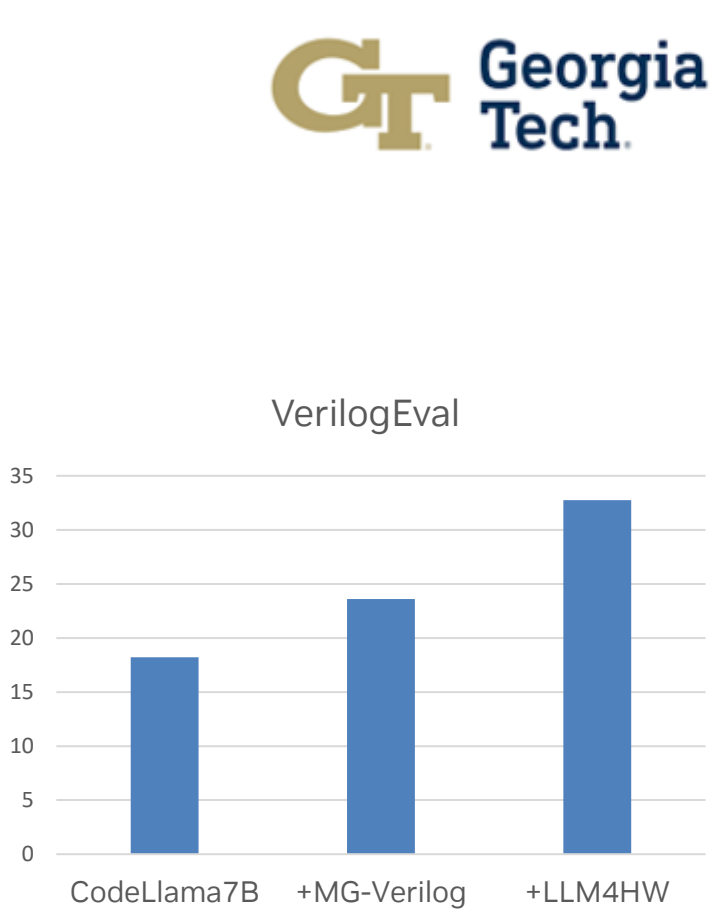
Lack of RTL and SVA training data on github



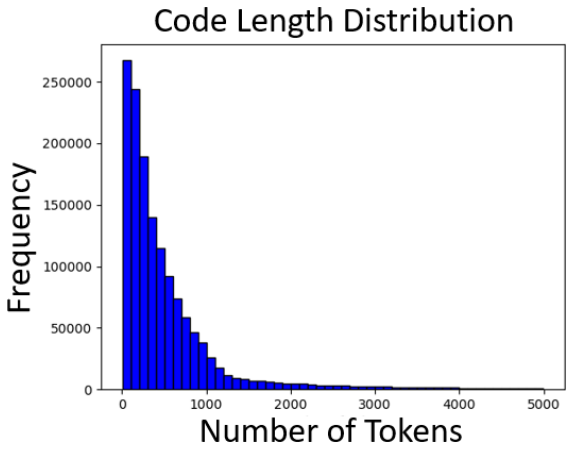
R. Li et al, StarCoder: may the source be with you!

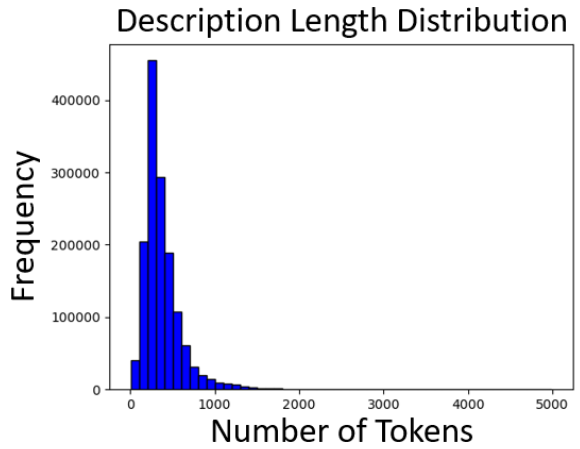
ICCAD Contest on LLM-Assisted Hardware Code Generation

(LLM4HWDesign @ ICCAD 2024)



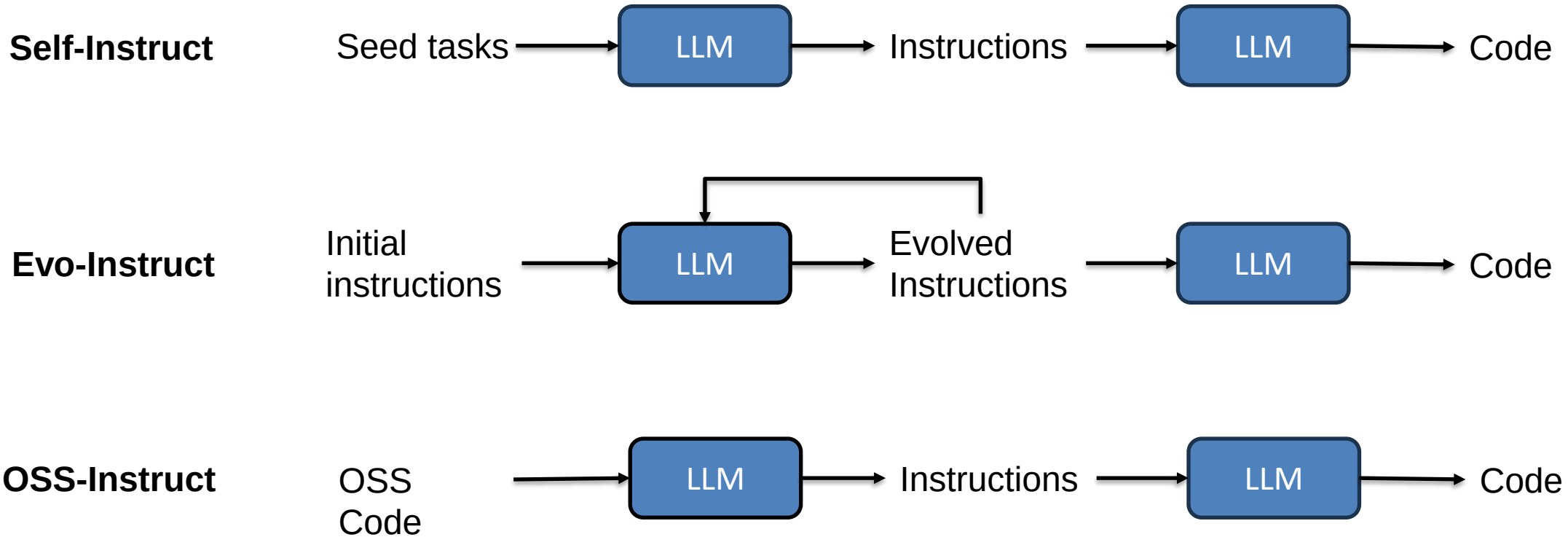
- **788,263** distinct data samples after deduplication
  - **> 60×** larger than current SOTA (11K in MG-Verilog, 8K in VerilogEval)
- Histogram on the length of data samples





Synthetic Data Generation

How to generate [instruction (specification) →code (Verilog)] pairs



Use NVIDIA nemotron-4-340b-instruct as LLM for SDG

CraftRTL: crafting SDG for VerilogEval

Conventional SDG:

80.1K x [Instruction → code]

Generate correct-by-construction data for non-textual problems such as : Karnaugh map, FSM and waveform:

28.5K x [non-textual problem instruction → code]

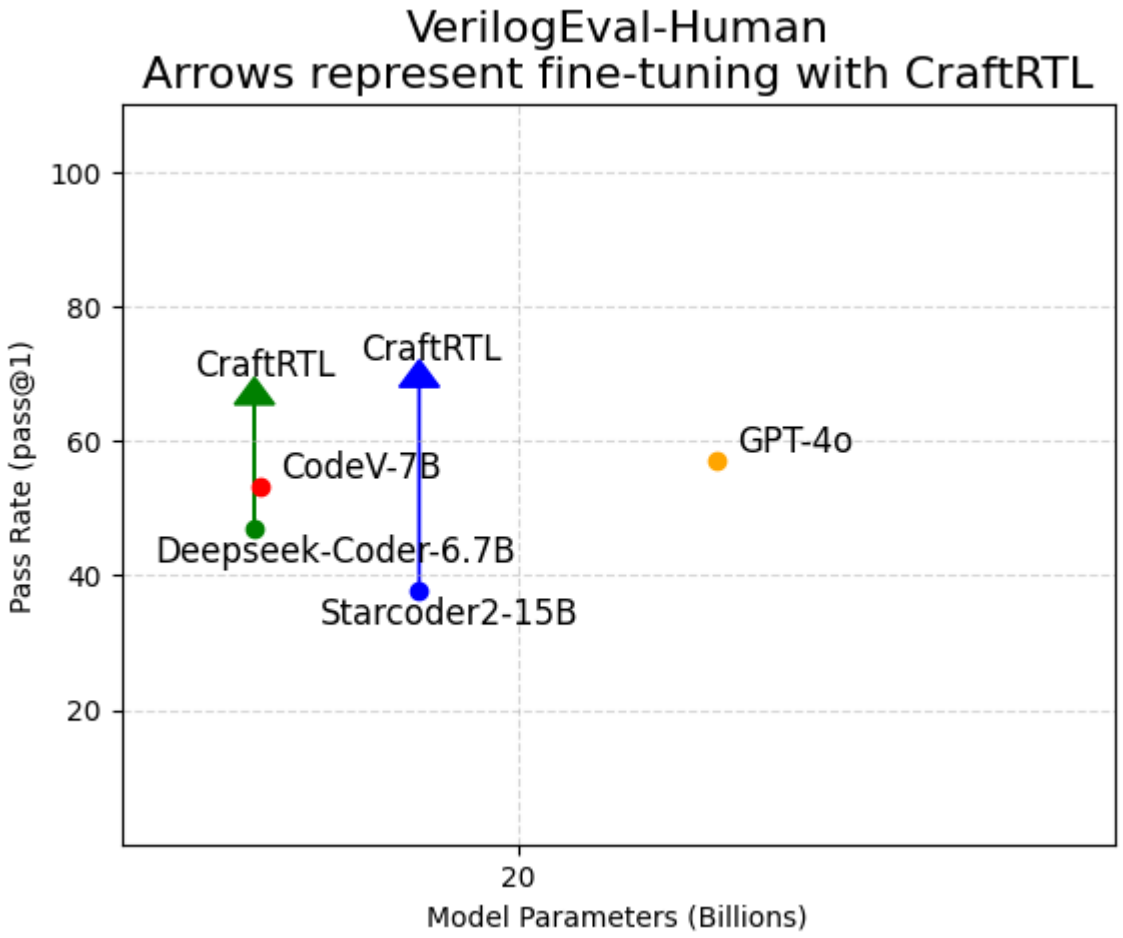
Generate error correction data by automatically analyzing common generation errors and injecting these errors to open-source code snippets:

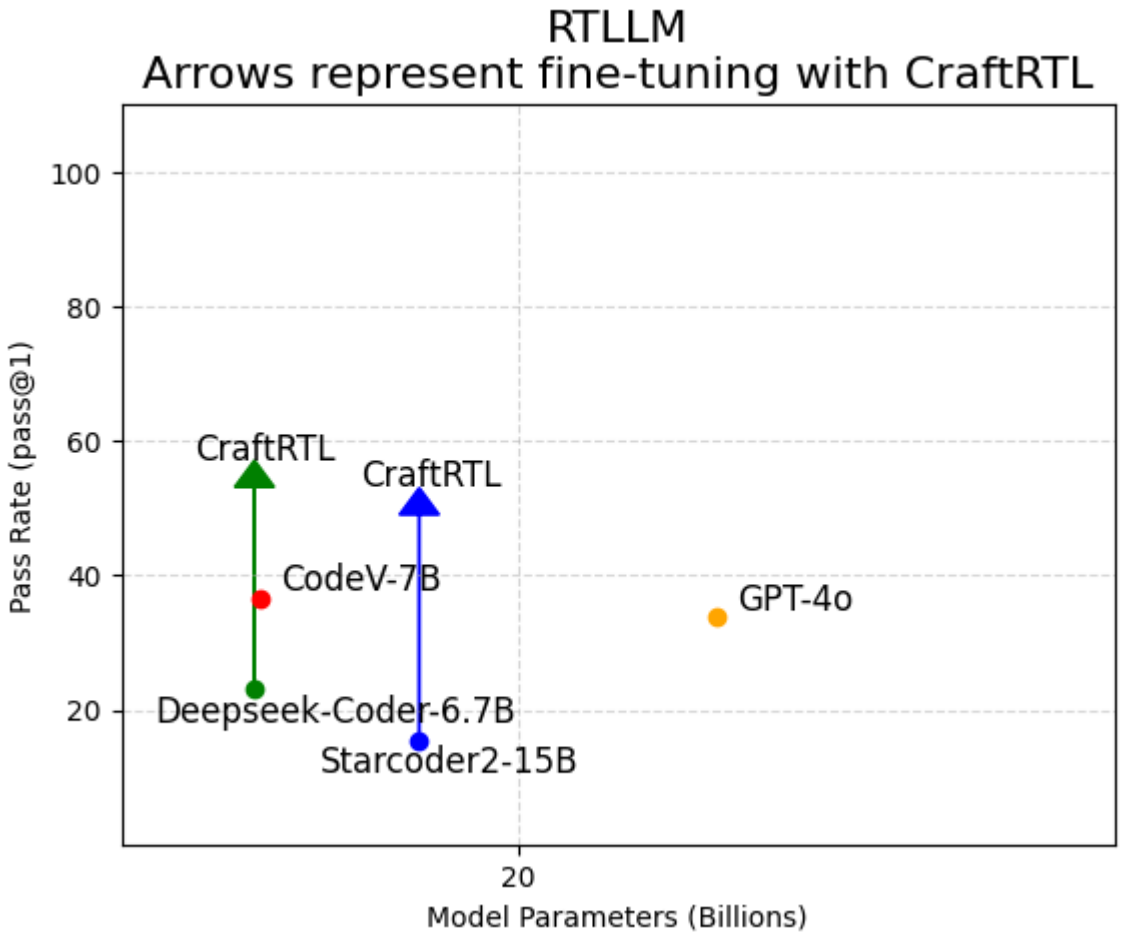
1.4K x [instruction, wrong code → correct code]

M. Liu and Y.-D. Tsai et al., CraftRTL: High-quality Synthetic Data Generation for Verilog Code Models with Correct-by-Construction Non-Textual Representations and Targeted Code Repair

NVIDIA

CraftRTL Models (7B/15B) are the new SOTA for Verilog Benchmarks

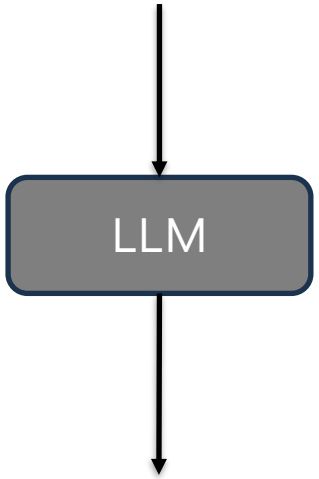




Agentic systems make LLMs much more powerful

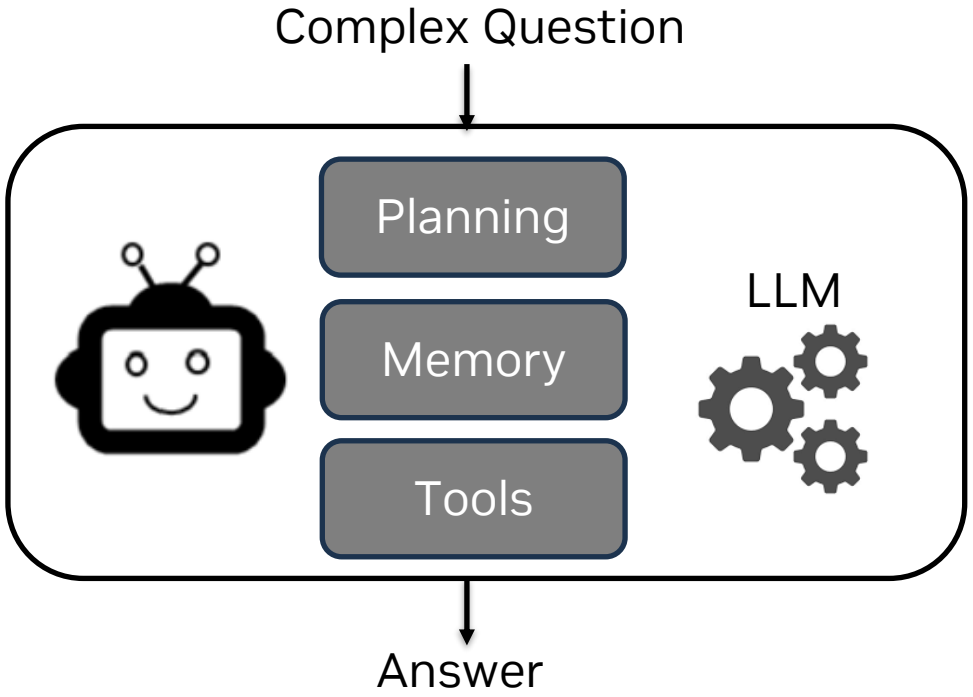
Non-Agentic

Simple Question

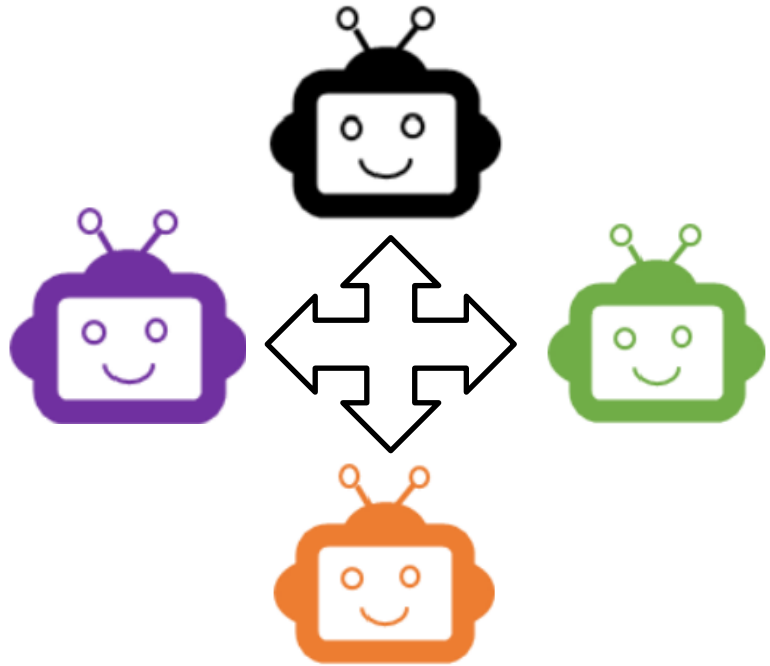


Answer

Agentic



Single Agent



Multi-Agent

NVIDIA

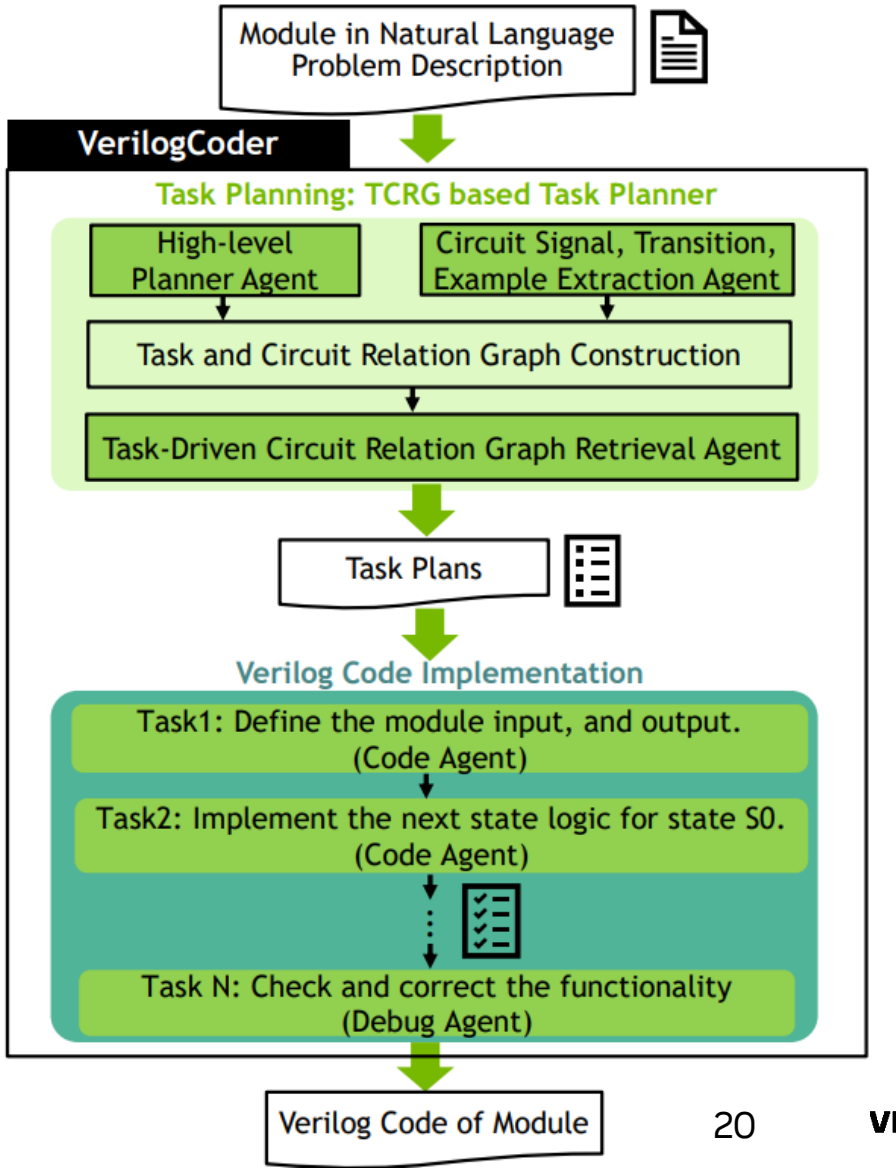
VerilogCoder: Solving VerilogEval benchmark

Two major stages:

The **task planning** stage generates a task plan

The **code implementation** stage writes code for each planned task and debug the code

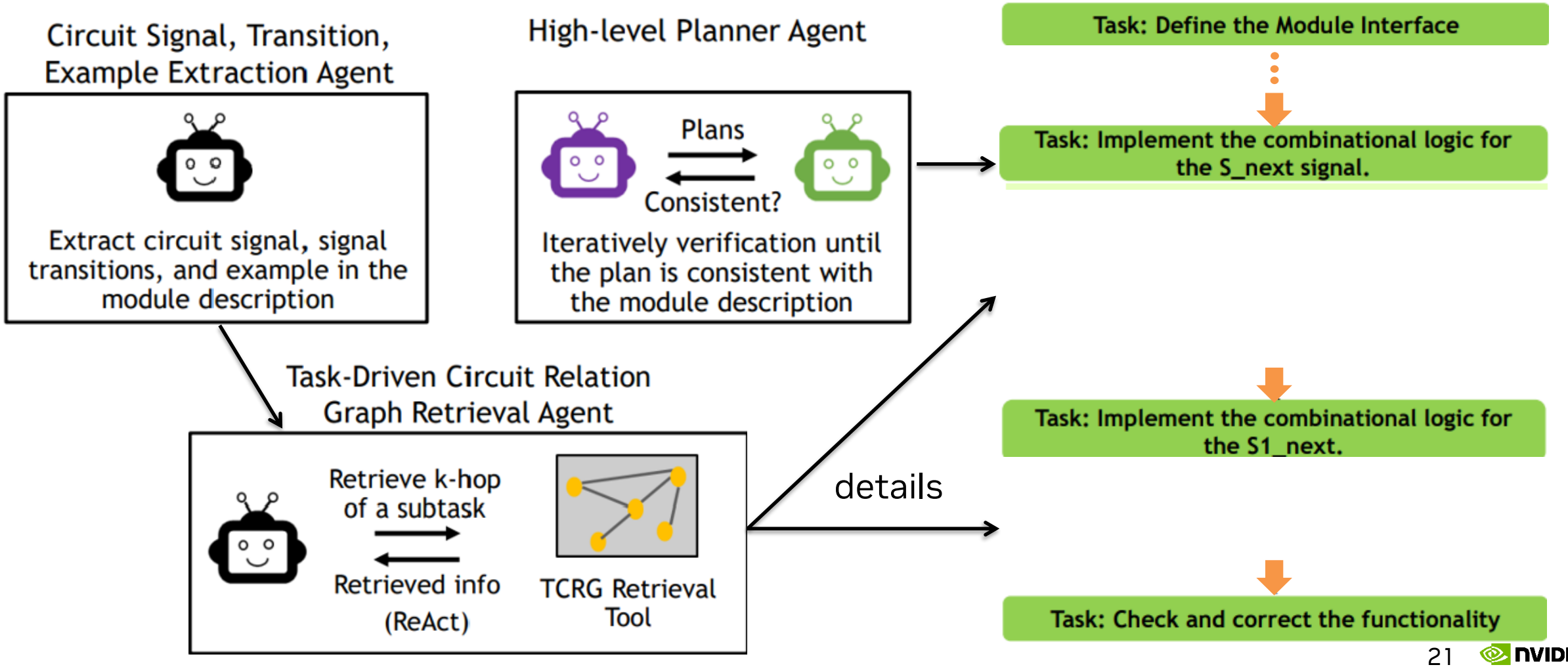
Dynamic task plan generation and execution



C.-T. Ho et al, VerilogCoder: Autonomous Verilog Coding Agents with Graph-based Planning and Abstract Syntax Tree (AST)-based Waveform Tracing Tool

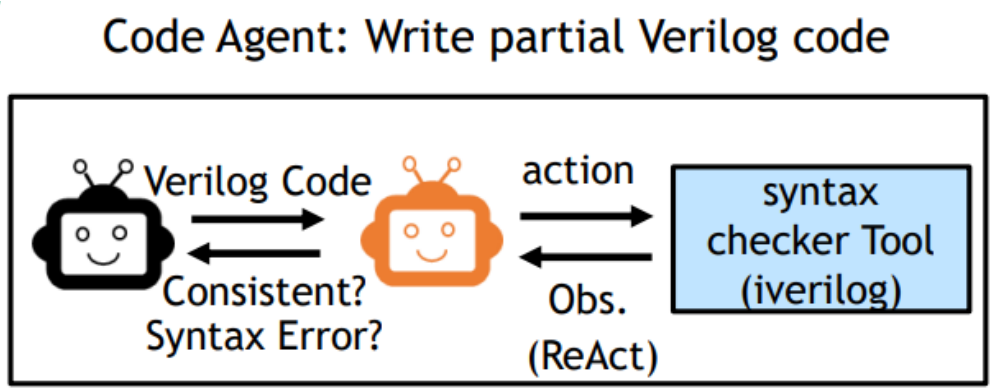
VIDIA.

Multi-agent planner builds Knowledge Graph to extract detailed information for task planning

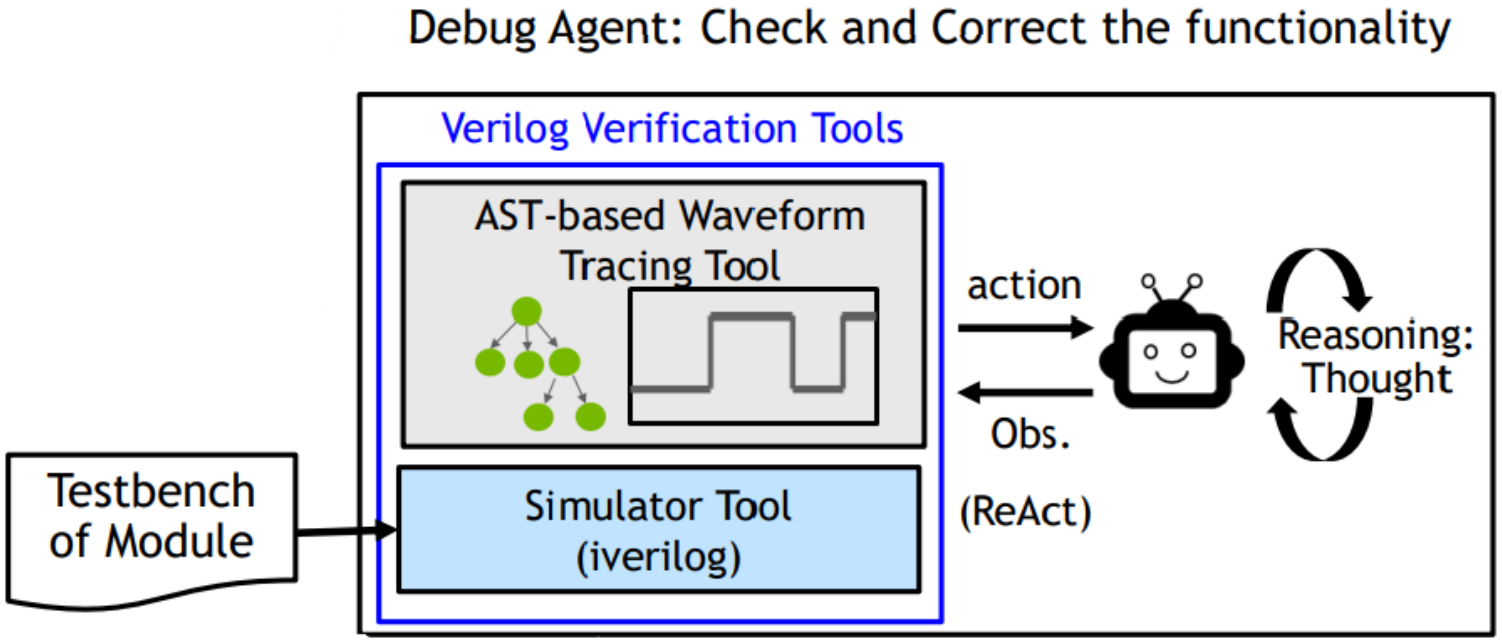


Multi-agent coder leverages syntax checker and waveform tracing tools to write and debug code

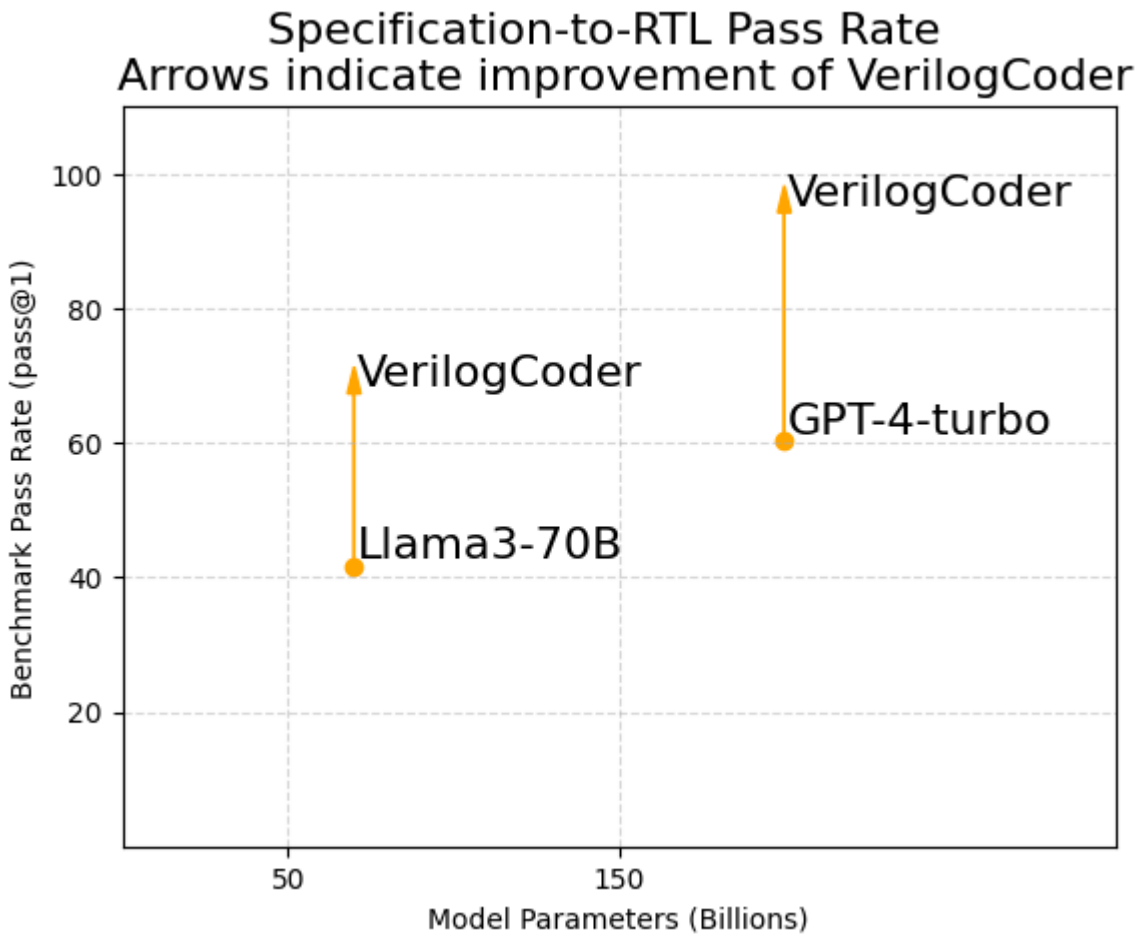
Writing code for each task



Debug final code



VerilogCoder solves most of problems in VerilogEval



Mate Terminal | Mate Terminal | Mate Terminal

```
(/home/scratch.chiatungh_nvresearch/nvcell_env/llm_env) bash-4.2$ python hardware_agent/examples/verilog_eval_agent/Verilog_eval_task_flow_ver1.py
reading  /home/scratch.chiatungh_nvresearch/hardware-agent-marco/hardware_agent/examples/verilog_testcases/verilog-eval-v2/dataset_dumpall/ Prob139_2013_q2bfsm_prompt.txt
reading  /home/scratch.chiatungh_nvresearch/hardware-agent-marco/hardware_agent/examples/verilog_testcases/verilog-eval-v2/dataset_dumpall/ Prob139_2013_q2bfsm_ref.sv
reading  /home/scratch.chiatungh_nvresearch/hardware-agent-marco/hardware_agent/examples/verilog_testcases/verilog-eval-v2/dataset_dumpall/ Prob139_2013_q2bfsm_test.sv
{'llm_class': <class 'adlrchat.langchain.LLMGatewayChat'>, 'llm_kwargs': {'model_name': 'gpt-4', 'temperature': 0.0, 'top_p': 1.0}}
case manager length =  1
current test is  2013_q2bfsm
current task id is  2013_q2bfsm
current test is  2013_q2bfsm
register tool  sequential_flipflop_latch_identify_tool <function sequential_flipflop_latch_identify_tool at 0x7f65abdac040>  caller:  <autogen.agentchat.assistant_agent.AssistantAgent object at 0x7f65a93ad1b0>  executer:  <autogen.agentc
hat.user_proxy_agent.UserProxyAgent object at 0x7f65a93ad390>
Hardware Agent Initialized 1 proxy, 0 rag proxy, 2 assistants
Hardware Agent Initialized 1 proxy, 0 rag proxy, 1 assistants
user (to chat_manager):

You are a Verilog RTL designer that can break down complicated implementation into subtasks implementation plans.

[Example Begin]

### Problem

I would like you to implement a module named TopModule with the following
interface. All input and output ports are one bit unless otherwise
specified.

 - input  clk
 - input  reset
 - input  in (8 bits)
 - output out (8 bits)

The module should implement an 8-bit registered incrementer. The 8-bit
input is first registered and then incremented by one on the next cycle.

Assume all sequential logic is triggered on the positive edge of the
clock. The reset input is active high synchronous and should reset the
output to zero.

### Solution

module TopModule
(
  input  logic       clk,
  input  logic       reset,
  input  logic [7:0] in,
  output logic [7:0] out
);

  // Sequential logic

  logic [7:0] reg_out;

  always @( posedge clk ) begin
    if ( reset )
      reg_out <= 0;
    else
      reg_out <= in;
  end

  // Combinational logic

  logic [7:0] temp_wire;

  always @(*) begin
    temp_wire = reg_out + 1;
```



Agenda

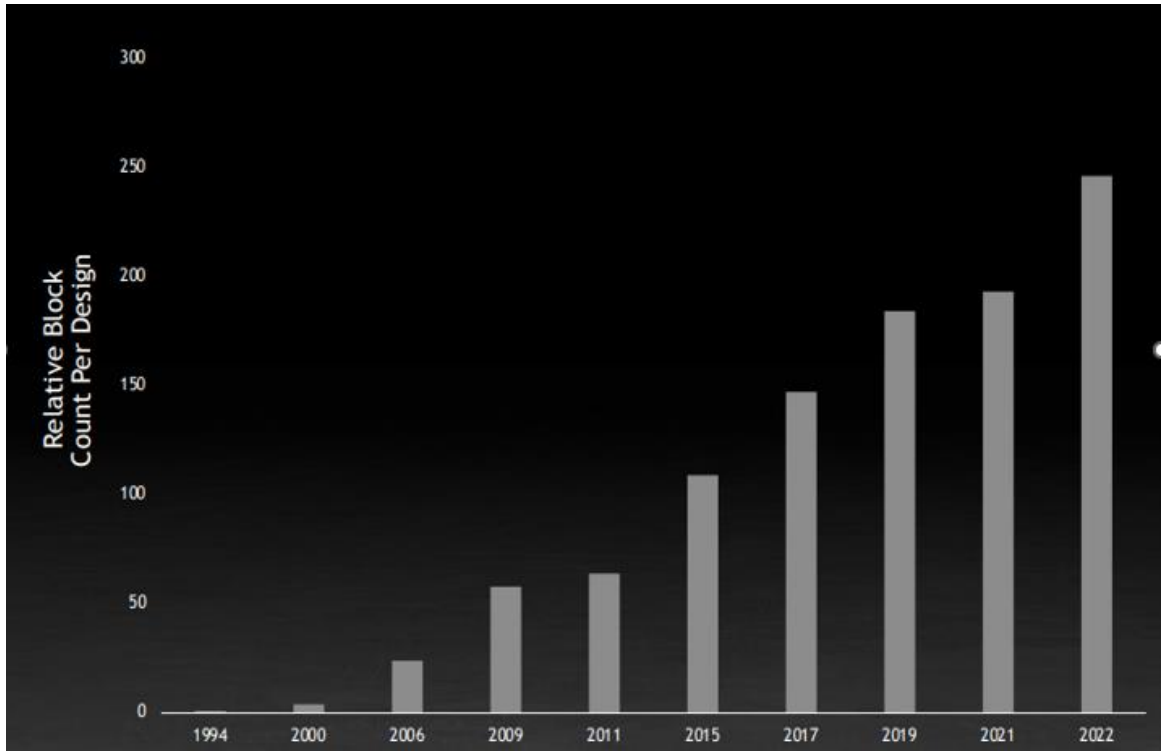
Introduction of AI for Chip Design

LLM for Verilog Generation

Research Opportunities

EDA Tools Scaling

"Average size of design levels off, unique block count skyrockets."



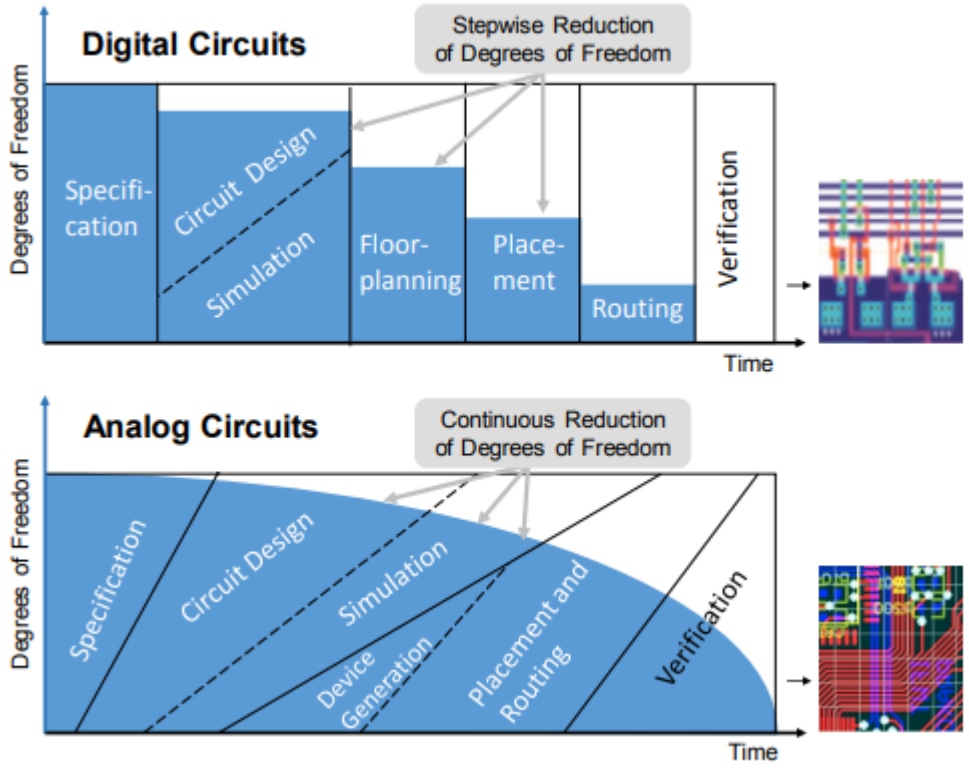
Sameer Halepete, NVIDIA, ISPD'22

- How can AI help?
  - AI-based design flow tuning
  - Scalable physical design tools based on GPU acceleration and AI
  - Placement, buffering, sizing, optimization, etc
  - RTL-level PPA optimization with LLM
  - AI-based PD problem triage

NVIDIA

Analog Design Automation

"Physical analog IC design has not been automated to the same degree as digital IC design. "

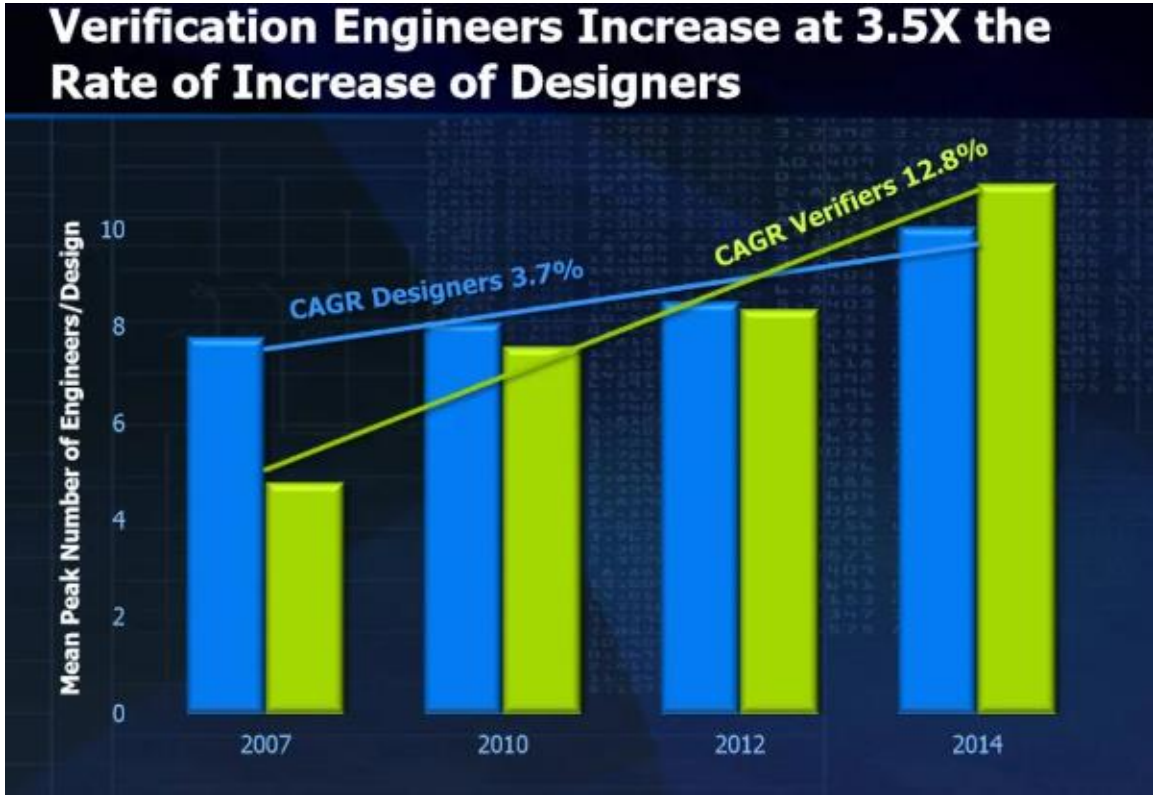


Scheible and Lienig, ISPD'15

- How can AI help?
  - Analog layout automation: constraint extraction and generation; layout optimization
  - Circuit optimization and generation
  - Simulation speedup
  - Verification

Verification

"Verification consumes more time and resources than design, and yet little headway is being made to optimize it" -- semiengineering.com



Rhines DVCON'16

- How can AI Help ?
  - Accelerate coverage closure with coverage prediction and stimulus optimization
  - Accelerate formal verification by learning to proof
  - Generate assertions with LLM
  - Improve debug productivity with AI agent

Formal Verification

Assist with test plan and assertion creation: generate test plan from spec and code; generate assertions from spec, code, and test plan

Assist with helper assertion: generates helper assertions to speed up the proof

Assist with abstraction creation: generate abstractions from code

Recommend verification strategy: analyzes which parts of systems are likely to contain errors and focuses the FV efforts on those areas, improving efficiency.

Improve FV tool efficiency: guides heuristics and reduces search state-space in the FV tools to improve efficiency

Assist with debug: automatic debug of FV errors

NVIDIA.

Solving the grand verification challenge: Debug

FV helps reduce the number of bugs, but bugs still exist

Debug is the most time-consuming activity in the verification process

Additional challenges to debug LLM-generated code

LLM-based debug assistant

Find and analyze relevant design specification, code, log, waveform, and bug report

Reason about the design logic

Generate hypothesis

Leverage models trained with existing documents and code

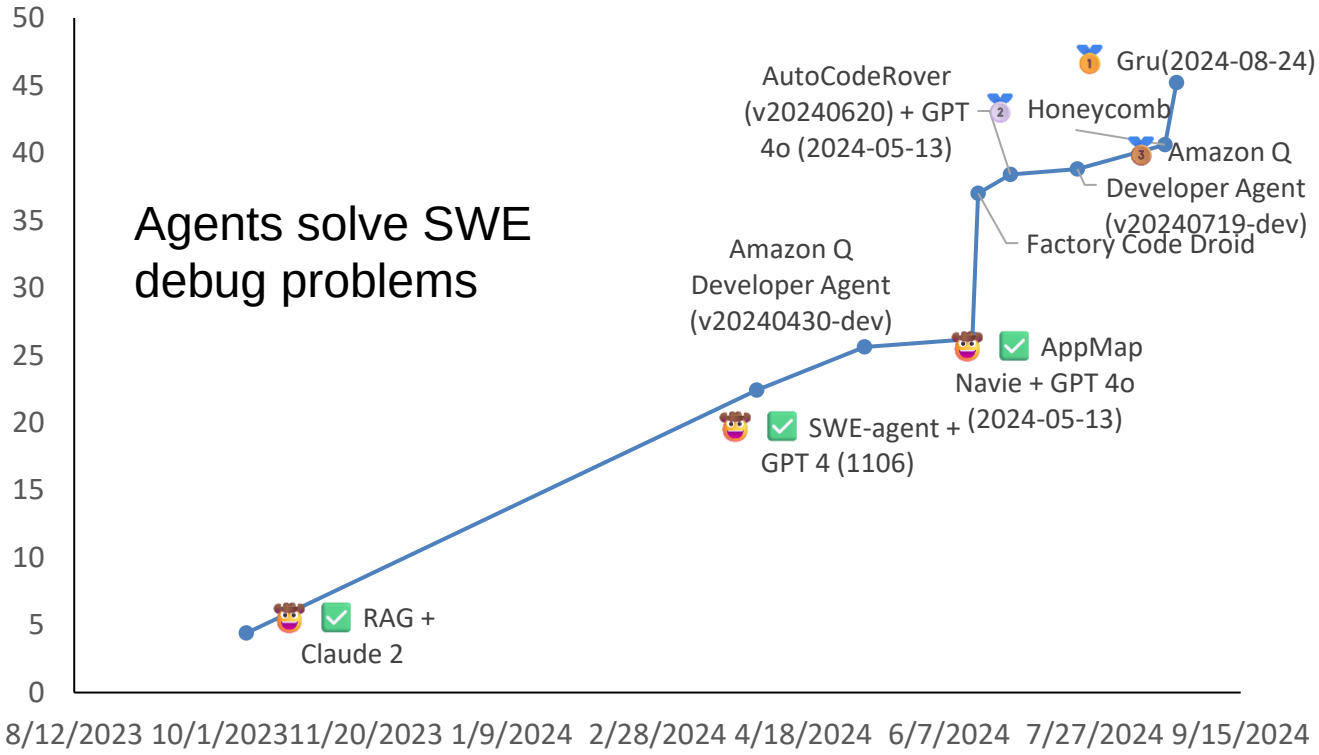
Leverage debug tools and FV

Agent-based approach for debug

SWE-Bench is a benchmark created from real github debug issues on popular repos like matplotlib

The latest Agent fixes almost half of the problems.

Can this extend to hardware?



https://www.swebench.com/

GPT-4o solves a real and unseen HW issue

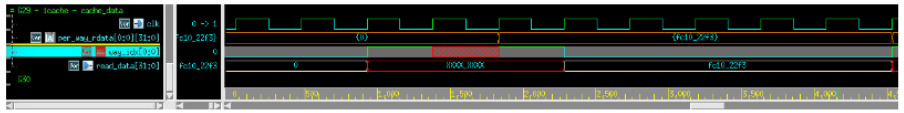
linuxbest commented on May 12 • edited

With the default setting, the NUM_WAYS of icache is 1, so the way_idx should always be 0, but it will change to 1. This behavior is acceptable with Verilator but fails with other simulators.

Verilator:



VCS:



```
597: -bank0 tag-fill: addr=0x80000000, way=1, blk_addr=0, tag_id=0x20000.            <<<< way=1
599: -bank0 mshr-dequeue: addr=0x80000000, id=0 (#0)
599: -bank0 mshr-pop: addr=0x80000000, tag=0x0, req_idx=0 (#0)
599: -bank0 data-fill: addr=0x80000000, way=1, blk_addr=0, data=0x2800bfff002930062900b1203031300000317f
601: -bank0 mshr-table
601: -bank0 tag-hit: addr=0x80000000, way=0, blk_addr=0, tag_id=0x20000 (#0)
603: -bank0 core-rd-rsp: addr=0x80000000, tag=0x0, req_idx=0, data=0xfc1022f3 (#0)
603: -bank0 data-read: addr=0x80000000, way=1, blk_addr=0, data=0xfc1022f3 (#0). <<<<<< way=1, it should
605: I$0 rsp: wid=0, PC=0x80000000, tmask=0001, instr=0xfc1022f3 (#0)
```

linuxbest commented on May 14 Author

The onehot encoder looks wrong to me.

When N==1, the encoder output data should be always 0.

Thanks.

```
diff --git a/hw/rtl/libs/VX_onehot_encoder.sv b/hw/rtl/libs/VX_onehot_encoder.sv
index 92c7d1ea..adaa4183 100644
--- a/hw/rtl/libs/VX_onehot_encoder.sv
+++ b/hw/rtl/libs/VX_onehot_encoder.sv
@@ -29,7 +29,7 @@ module VX_onehot_encoder #(
 );
     if (N == 1) begin

-        assign data_out  = data_in;
+        assign data_out  = 0;
         assign valid_out = data_in;

     end else if (N == 2) begin
```

The issue you're describing seems related to how the `data_out` value is computed for an instruction cache (icache) with `NUM_WAYS = 1`. In this case, `way_idx` should always be 0, but it is changing to 1. This could happen if the logic calculating `data_out` or `valid_out` is not correctly handling the case where `N == 1`.

To fix this issue, we can enforce that when `N == 1`, the `data_out` and `valid_out` behave consistently, ensuring that the output is always zero for `way_idx` regardless of other conditions. Here's the corrected module:

```
if (N == 1) begin

    assign data_out  = 0; // Ensure way_idx is always 0 for N=1
    assign valid_out = data_in;
```

Key Changes:

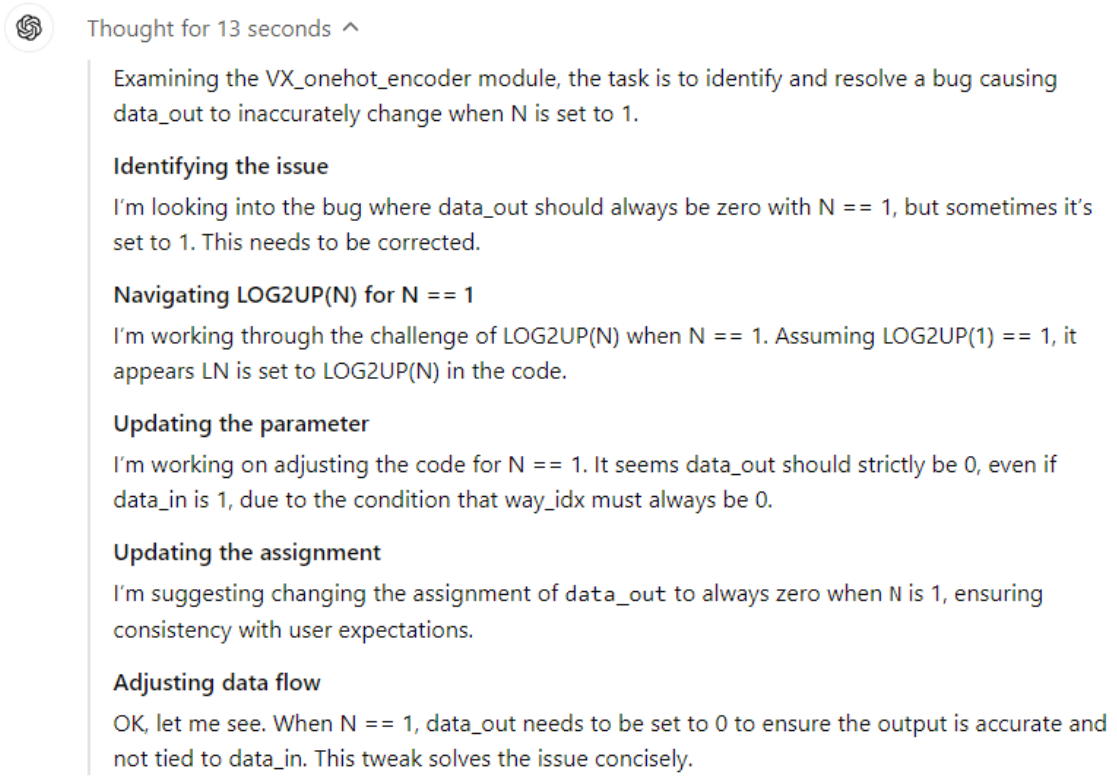
- For `N == 1`, `data_out` is explicitly set to `0`, ensuring that `way_idx` remains `0`.
- This guarantees the correct behavior when `NUM_WAYS == 1`, avoiding any potential bugs in handling this special case.

This should resolve the issue where `way_idx` unexpectedly changes to 1 when it should always remain 0 in this configuration.

https://github.com/vortexgpgpu/vortex/issues/126

AI learns to reason deeply about chip design

Static plan (predefined CoT, taskflow) work but not flexible
Dynamic plan ('think step by step', VerilogCoder task plan) are more desirable
O1 improves this capability by learning – learning to reason, only good at math/science
Future: Train domain-specific LLM or leverage general-purpose model?



"o1 learns to hone its chain of thought and refine the strategies it uses. It learns to recognize and correct its mistakes. It learns to break down tricky steps into simpler ones. It learns to try a different approach when the current one isn't working. "

OpenAI, Learning to Reason with LLMs

IEEE International Conference on LLM-Aided Design, 2025

Time: June 26-27, 2025

Location: Stanford University, Stanford, CA



The main theme of LAD this year will revolve around **agentic optimization and scaling inference-time methods**, but we welcome a broad range of topics on new methodologies, tools, datasets, and benchmarks pertaining to:

- *Agentic workflows for design automation and optimization*
- *Inference-time techniques for design*
- *LLM-aided HW/SW design, code generation, and test plan generation*
- *System-level design methodology development with LLMs*
- *Finetuning of large foundation models for specialization in design automation*
- *New datasets and benchmarks of relevance to LLM-aided design*
- *Evaluation and verification of LLM-aided designs*
- *LLM-aided design for software development, IT automation, site reliability, and regulatory compliance.*
- *LLMs for EDA, including RTL, HLS, physical design, and EDA scripting*
- *LLMs for reasoning and logic used in design process*
- *Computational efficiency of LLM-aided design tools*
- *Data science and data analytics for LLM-aided design*
- *Security of LLM-generated designs*
- *Privacy, copyright, and other regulatory concerns around LLM-aided design*
- *LLM-aided bug-fixing*
- *LLM-aided design for various application domains, such as 3D manufacturing, material discovery, sustainability, etc.*

Conclusion

- AI helps chip design in analysis, optimization and assistance tasks
- Fine-tuned model can achieve state-of-the-art results with a fraction of the inference cost with carefully designed synthetic data
- Agentic LLM systems can significantly boost code generation accuracy
- verification, EDA tools scaling, and analog design automation are major research opportunities for AI today

NVIDIA

