Laying the Foundations for LLMs in Open-source EDA

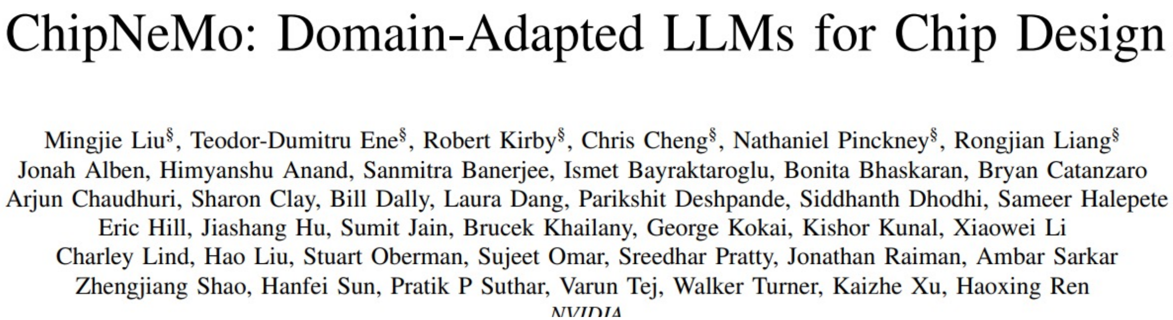
Presented by: Vidya A. Chhabria

Arizona State University

NEW YORK UNIVERSITY

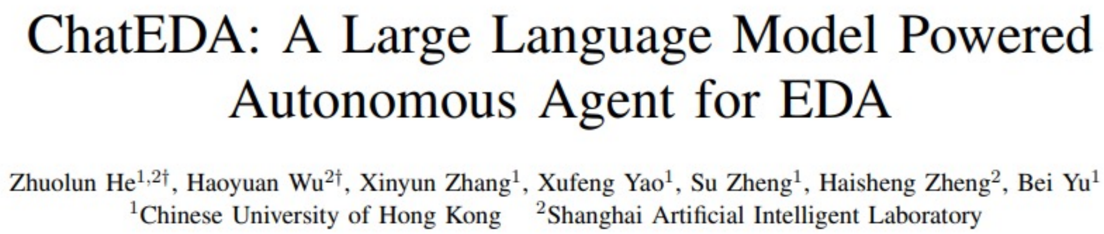
LLMs for physical design?

- **Challenges with physical design:**
  - Proprietary tools
  - Proprietary PDKs
  - Closed-source APIs
- **Limited existing efforts:**



ChipNeMo: Domain-Adapted LLMs for Chip Design

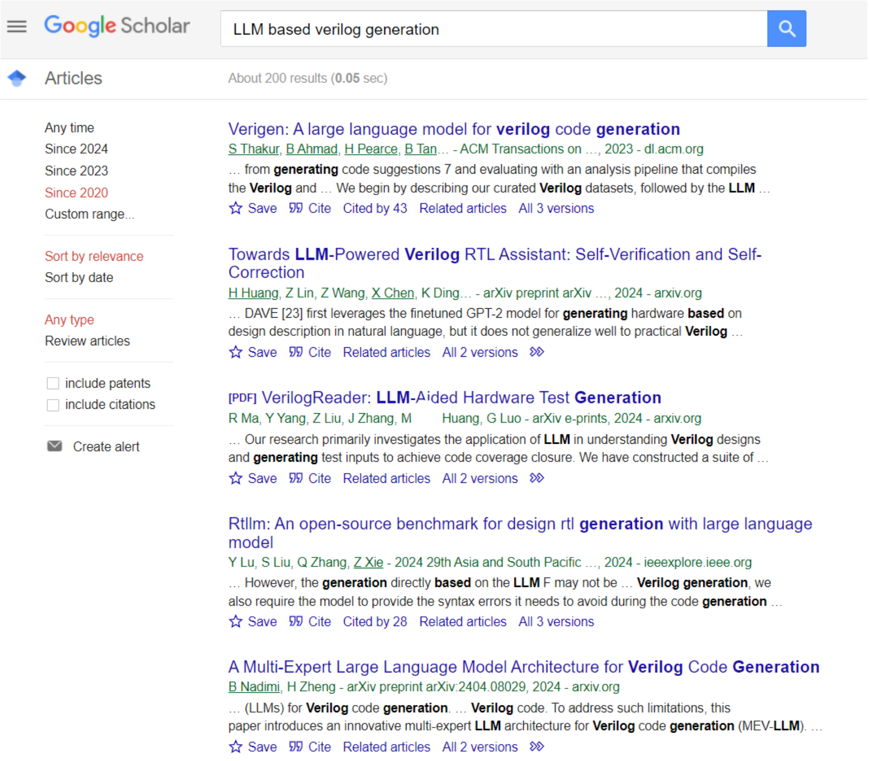
Mingjie Liu[§], Teodor-Dumitru Ene[§], Robert Kirby[§], Chris Cheng[§], Nathaniel Pinckney[§], Rongjian Liang[§]
Jonah Alben, Himyanshu Anand, Sanmitra Banerjee, Ismet Bayraktaroglu, Bonita Bhaskaran, Bryan Catanzaro
Arjun Chaudhuri, Sharon Clay, Bill Dally, Laura Dang, Parikshit Deshpande, Siddhanth Dhodhi, Sameer Halepete
Eric Hill, Jiashang Hu, Sumit Jain, Brucek Khailany, George Kokai, Kishor Kunal, Xiaowei Li
Charley Lind, Hao Liu, Stuart Oberman, Sujeet Omar, Sreedhar Pratty, Jonathan Raiman, Ambar Sarkar
Zhengjiang Shao, Hanfei Sun, Pratik P Suthar, Varun Tej, Walker Turner, Kaizhe Xu, Haoxing Ren
NVIDIA



ChatEDA: A Large Language Model Powered Autonomous Agent for EDA

Zhuolun He[1,2†], Haoyuan Wu[2†], Xinyun Zhang[1], Xufeng Yao[1], Su Zheng[1], Haisheng Zheng[2], Bei Yu[1]
[1]Chinese University of Hong Kong [2]Shanghai Artificial Intelligent Laboratory

- **LLMs for RTL code generation:**
  - Language is standardized
  - Some-level of data curated (open cores, RTLCoder, VGen)



LLM based verilog generation

About 200 results (**0.05** sec)

Verigen: A large language model for **verilog** code **generation**
S Thakur, B Ahmad, H Pearce, B Tan... - ACM Transactions on ..., 2023 - dl.acm.org
... from **generating** code suggestions 7 and evaluating with an analysis pipeline that compiles the **Verilog** and ... We begin by describing our curated **Verilog** datasets, followed by the **LLM** ...
☆ Save 99 Cite Cited by 43 Related articles All 3 versions

Towards **LLM**-Powered **Verilog** RTL Assistant: Self-Verification and Self-Correction
H Huang, Z Lin, Z Wang, X Chen, K Ding... - arXiv preprint arXiv ..., 2024 - arxiv.org
... DAVE [23] first leverages the finetuned GPT-2 model for **generating** hardware **based** on design description in natural language, but it does not generalize well to practical **Verilog** ...
☆ Save 99 Cite Related articles All 2 versions

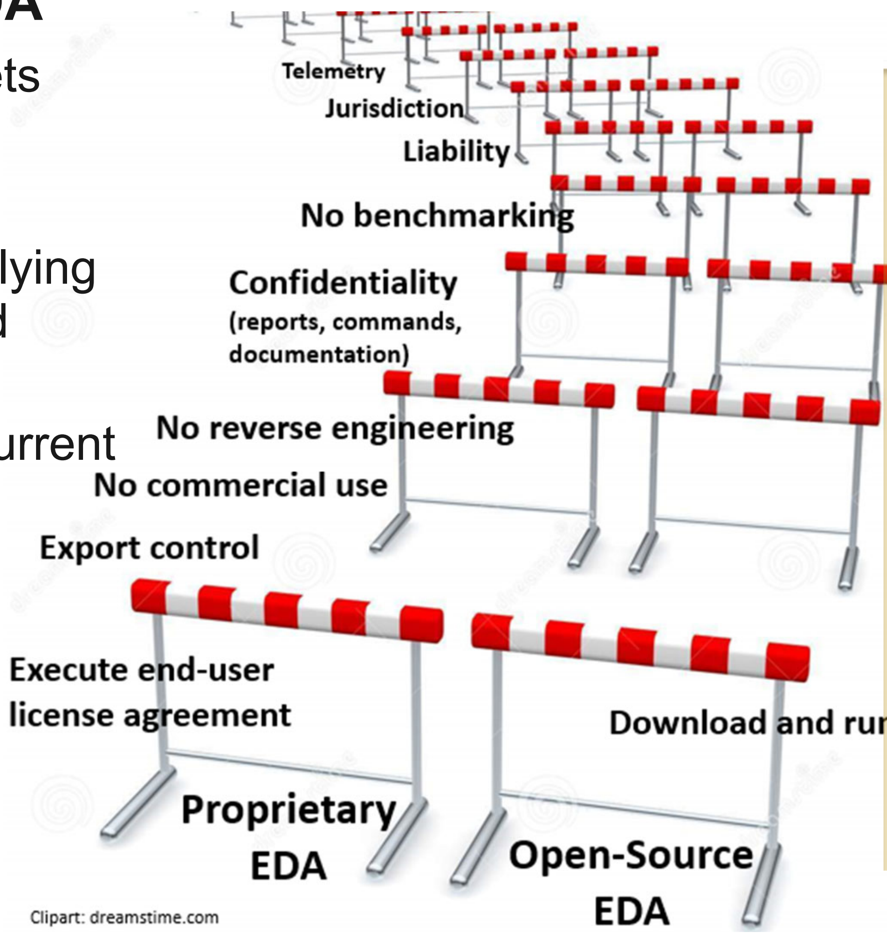
[PDF] VerilogReader: **LLM**-Aided Hardware Test **Generation**
R Ma, Y Yang, Z Liu, J Zhang, M Huang, G Luo - arXiv e-prints, 2024 - arxiv.org
... Our research primarily investigates the application of **LLM** in understanding **Verilog** designs and **generating** test inputs to achieve code coverage closure. We have constructed a suite of ...
☆ Save 99 Cite Related articles All 2 versions

Rtllm: An open-source benchmark for design rtl **generation** with large language model
Y Lu, S Liu, Q Zhang, Z Xie - 2024 29th Asia and South Pacific ..., 2024 - ieeexplore.ieee.org
... However, the **generation** directly **based** on the **LLM** F may not be ... **Verilog generation**, we also require the model to provide the syntax errors it needs to avoid during the code **generation** ...
☆ Save 99 Cite Cited by 28 Related articles All 3 versions

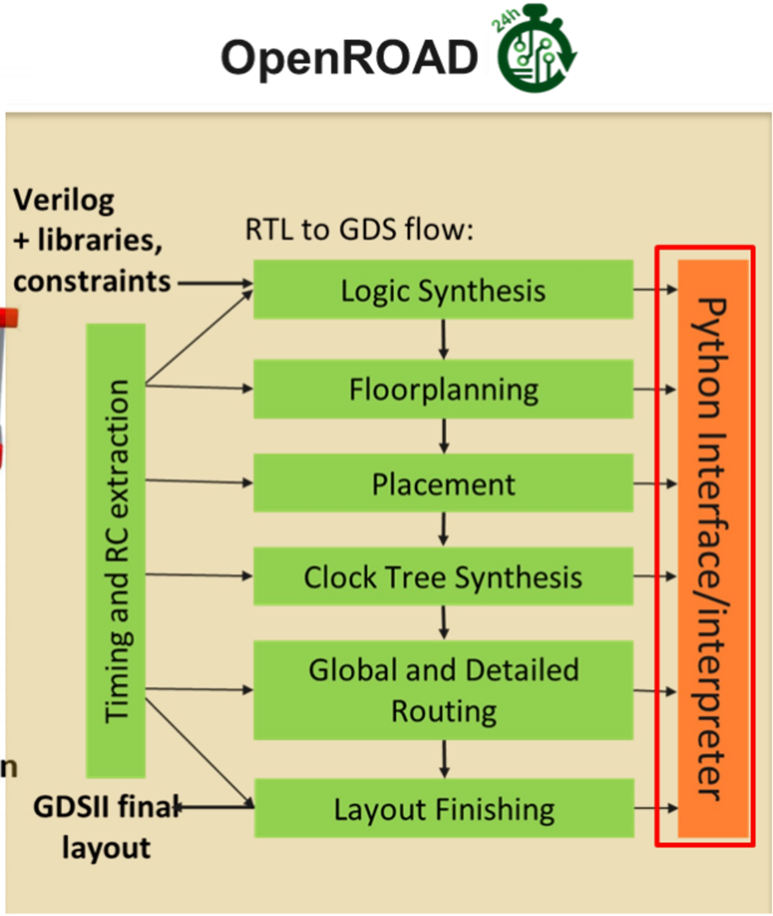
A Multi-Expert Large Language Model Architecture for **Verilog** Code **Generation**
B Nadimi, H Zheng - arXiv preprint arXiv:2404.08029, 2024 - arxiv.org
... (LLMs) for **Verilog** code **generation**. ... **Verilog** code. To address such limitations, this paper introduces an innovative multi-expert **LLM** architecture for **Verilog** code **generation** (MEV-**LLM**). ...
☆ Save 99 Cite Related articles All 2 versions

Open-source tools for physical design

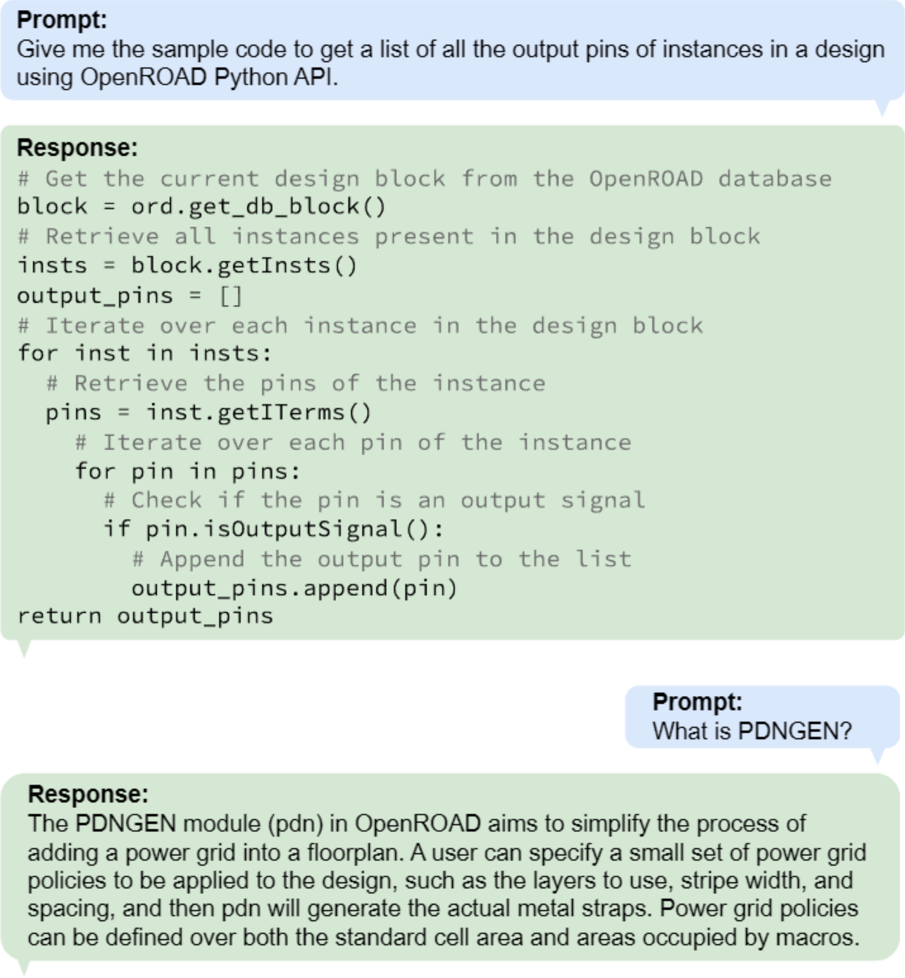
- **OpenROAD helps overcome challenges with proprietary EDA**
  - Allows the development of datasets
  - In built support for Python APIs allowing easy interaction with ML
  - Access to source code and underlying APIs for tool script generation and training
  - Python interfaces are easier for current ML techniques compared to TCL
- **Remaining challenges**
  - Documentation (Needs community driven effort)
  - Continuous AI specific API development



Clipart: dreamstime.com



EDA Corpus: An LLM dataset for OpenROAD



Prompt:
Give me the sample code to get a list of all the output pins of instances in a design using OpenROAD Python API.

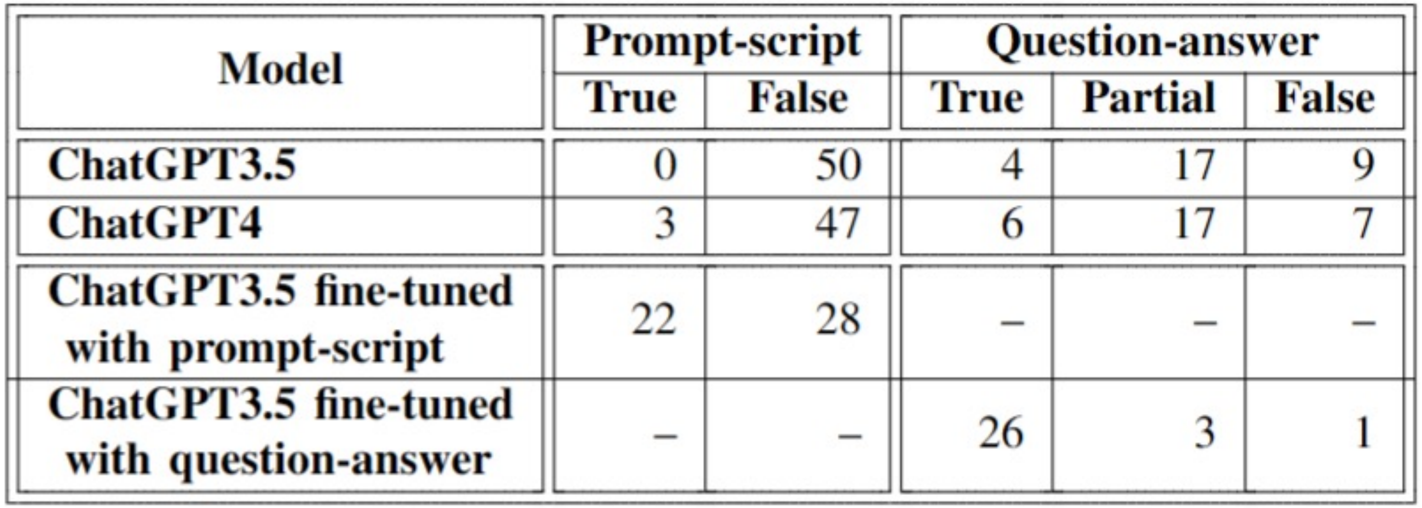
Response:

```
# Get the current design block from the OpenROAD database
block = ord.get_db_block()
# Retrieve all instances present in the design block
insts = block.getInsts()
output_pins = []
# Iterate over each instance in the design block
for inst in insts:
  # Retrieve the pins of the instance
  pins = inst.getITerms()
    # Iterate over each pin of the instance
    for pin in pins:
      # Check if the pin is an output signal
      if pin.isOutputSignal():
        # Append the output pin to the list
        output_pins.append(pin)
return output_pins
```

Prompt:
What is PDNGEN?

Response:
The PDNGEN module (pdn) in OpenROAD aims to simplify the process of adding a power grid into a floorplan. A user can specify a small set of power grid policies to be applied to the design, such as the layers to use, stripe width, and spacing, and then pdn will generate the actual metal straps. Power grid policies can be defined over both the standard cell area and areas occupied by macros.

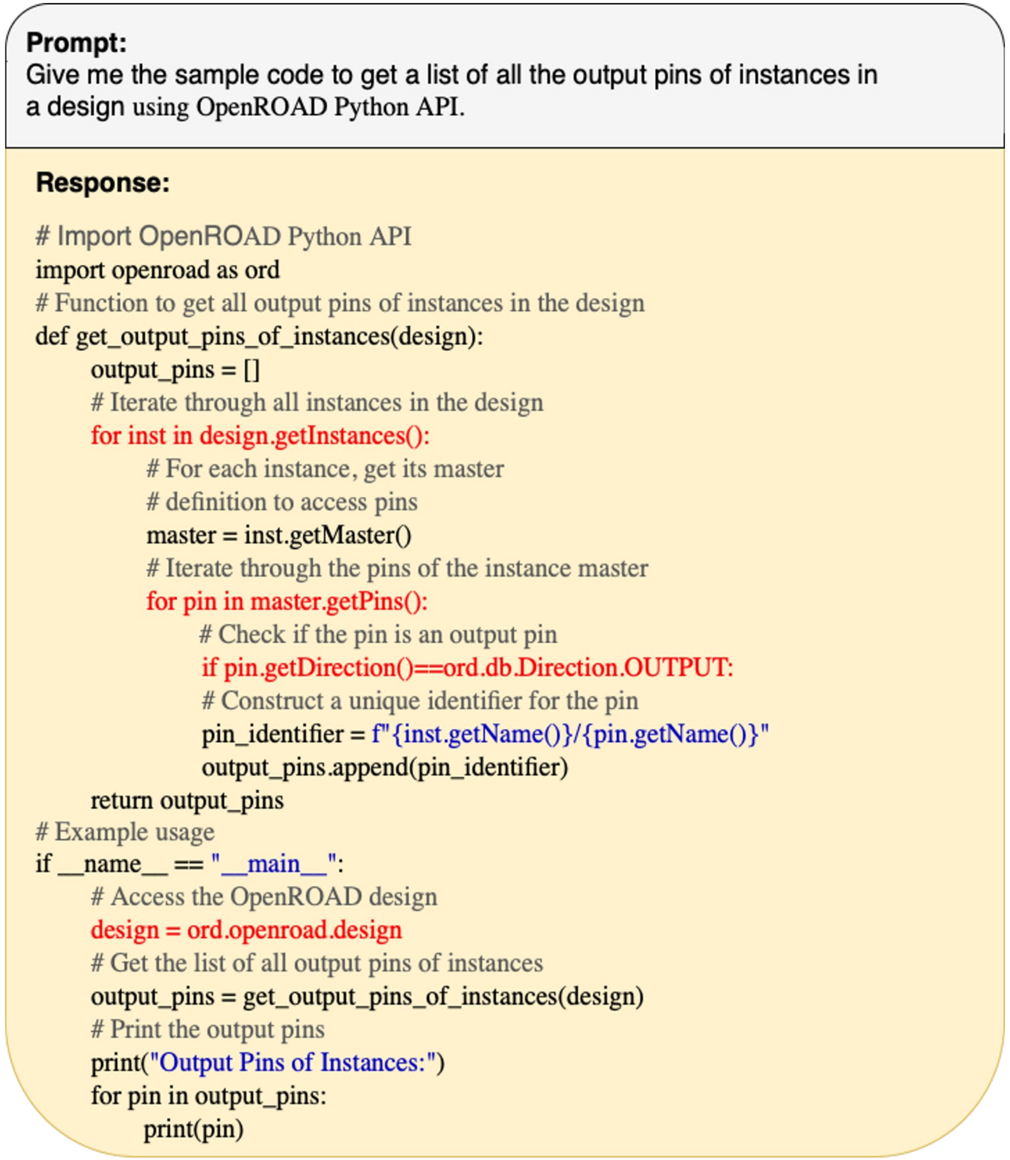
- EDA Corpus: A Large Language Model Dataset for Enhanced Interaction with OpenROAD https://github.com/OpenROAD-Assistant/EDA-Corpus
- Over 1500 datapoints of prompt-response pairs including prompt-scripts (Python) and question-answers for various physical design tasks
- Fine-tuning ChatGPT3.5 with EDA Corpus:



Model	Prompt-script		Question-answer		
	True	False	True	Partial	False
ChatGPT3.5	0	50	4	17	9
ChatGPT4	3	47	6	17	7
ChatGPT3.5 fine-tuned with prompt-script	22	28	–	–	–
ChatGPT3.5 fine-tuned with question-answer	–	–	26	3	1

B.-Y. Wu, U. Sharma et. al., EDA Corpus, LAD’24

LLM finetuning examples



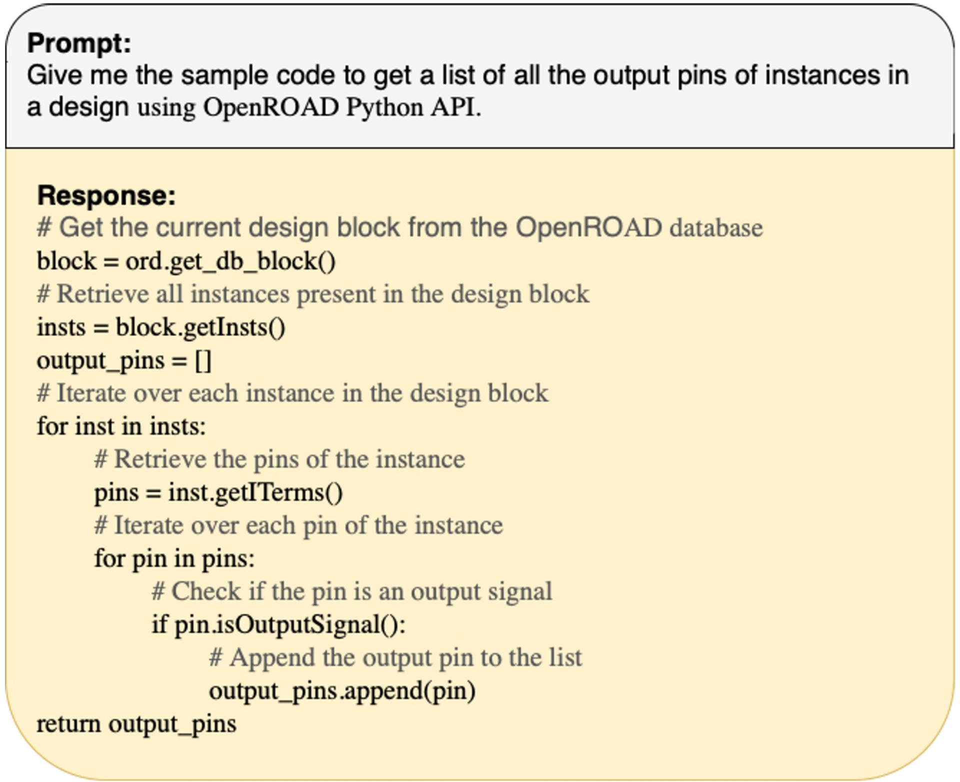
Prompt:
Give me the sample code to get a list of all the output pins of instances in a design using OpenROAD Python API.

Response:

```
# Import OpenROAD Python API
import openroad as ord
# Function to get all output pins of instances in the design
def get_output_pins_of_instances(design):
    output_pins = []
    # Iterate through all instances in the design
    for inst in design.getInstances():
        # For each instance, get its master
        # definition to access pins
        master = inst.getMaster()
        # Iterate through the pins of the instance master
        for pin in master.getPins():
            # Check if the pin is an output pin
            if pin.getDirection()==ord.db.Direction.OUTPUT:
                # Construct a unique identifier for the pin
                pin_identifier = f"{inst.getName()}/{pin.getName()}"
                output_pins.append(pin_identifier)
    return output_pins
# Example usage
if __name__ == "__main__":
    # Access the OpenROAD design
    design = ord.openroad.design
    # Get the list of all output pins of instances
    output_pins = get_output_pins_of_instances(design)
    # Print the output pins
    print("Output Pins of Instances:")
    for pin in output_pins:
        print(pin)
```

- Domain-specific datasets allow for improved EDA tool scripting using existing foundation models

EDA Corpus Fine-Tuning on ChatGPT 3.5



Prompt:
Give me the sample code to get a list of all the output pins of instances in a design using OpenROAD Python API.

Response:

```
# Get the current design block from the OpenROAD database
block = ord.get_db_block()
# Retrieve all instances present in the design block
insts = block.getInsts()
output_pins = []
# Iterate over each instance in the design block
for inst in insts:
    # Retrieve the pins of the instance
    pins = inst.getITerms()
    # Iterate over each pin of the instance
    for pin in pins:
        # Check if the pin is an output signal
        if pin.isOutputSignal():
            # Append the output pin to the list
            output_pins.append(pin)
return output_pins
```

Conclusion

- **Upcoming work: OpenROAD-Assistant**
  - Using EDA Corpus we are nearly done with building an open model that we will hopefully share with the public built on Llama3 and RAG-based training.
- **Open-datasets, tools, and PDKs are crucial for LLM research:**
  - 1500 datapoint is still VERY small
  - Scaling and maintaining the repository requires a collaborative effort

Thank you!!

Any Questions? Comments? Concerns?