Introduction to Special Issue on Large Language Models for Electronic System Design Automation

ROBERT PAUL DICK, Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, United States
HAMMOND PEARCE, School of Computer Science and Engineering, University of New South Wales Faculty of Engineering, Sidney, Australia
LI SHANG, School of Computer Science, Fudan University, Shanghai, China
FAN YANG, School of Microelectronics, Fudan University, Shanghai, China

Large Language Models are having a substantial impact on electronic design automation in areas ranging from hardware architecture to verification and optimization. The special issue provides a snapshot of work on this topic. This introduction describes and provides context for the research area, describes the organization of the special issue, and provides terse summaries of each of its papers.

CCS Concepts: • **Hardware** → **Electronic design automation**; • **Computing methodologies** → *Natural language processing;*

Additional Key Words and Phrases: Design automation, large language models, artificial intelligence, machine learning, hardware, software

ACM Reference Format:
Robert Paul Dick, Hammond Pearce, Li Shang, and Fan Yang. 2025. Introduction to Special Issue on Large Language Models for Electronic System Design Automation. *ACM Trans. Des. Autom. Electron. Syst.* 30, 6, Article 89 (October 2025), 3 pages. https://doi.org/10.1145/3746636

A key difference between electronics and most other areas of human endeavor is that, very early in its history, advances in electronics became dependent on automation. Electronic design automation (EDA) has played a key role in the rapid advances in electronic system capabilities, with early efforts focused on lower-level aspects of the design process and current efforts considering everything from physical design to system-level design. EDA enables a feedback loop in which computers are used to design more complex and capable computers.

Recent advances in artificial intelligence have enabled new capabilities once possessed only by humans; large language models (LLMs) provide a compelling example. Based on transformer-type deep neural networks, LLMs are trained over vast quantities of natural language prose and code which they can learn to predict or mimic, enabling them to carry out instructions in the context of

Authors' Contact Information: Robert Paul Dick, Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, Michigan, United States; e-mail: dickrp@umich.edu; Hammond Pearce, School of Computer Science and Engineering, University of New South Wales Faculty of Engineering, Sidney, New South Wales, Australia; e-mail: hammond.pearce@unsw.edu.au; Li Shang, School of Computer Science, Fudan University, Shanghai, Shanghai, China; e-mail: li.shang@fudan.edu.cn; Fan Yang, School of Microelectronics, Fudan University, Shanghai, Shanghai, China; e-mail: fan.yang@fudan.edu.cn.

ACM 1084-4309/2025/10-ART89
https://doi.org/10.1145/3746636

their training. The largest models—which are now trained on a substantial portion of all publicly available textually recorded human knowledge—are capable of using knowledge of a vast number of concepts when following instructions and answering questions. Of particular relevance to this special issue, they can complete tasks relevant to EDA based on hardware and software design documents and source code.

Academic and industry researchers have recently started using LLMs to address a very broad range of EDA problems, including automating both pre- and post-silicon steps in the design and test process; assisting with improving design efficiency and power/performance/area metrics; and debugging and fault root cause analysis, which are particularly useful given the large proportion of design time spent on validation and verification.

The purpose of this special issue is to provide readers with a snapshot of the current state of research on using LLMs to solve a broad range of EDA problems, spanning from physical level to system level. The current trajectory of research in this new field suggests that it will grow rapidly in the next few years.

The special issue is divided into four sections.

The first section focuses on **surveys, benchmarks, and benchmarking**, which are important for understanding the evolving EDA capabilities of LLMs. The first paper provides a taxonomy and survey of research on using LLMs in EDA. The second provides a benchmarking suite for use in evaluating LLMs for Verilog hardware description language (HDL) synthesis tasks and evaluates several open and closed models using the suite. The third evaluates 11 LLMs on Verilog HDL synthesis tasks of varying complexities and provides guidelines for using LLMs for hardware synthesis.

The second section focuses on using **LLMs for hardware generation**. The first paper describes a method of generating register transfer level (RTL) Verilog LLM training data for HDL code generation and methods to guide an LLM toward power-, performance-, and area-optimized designs. The second describes an LLM-based analog and mixed-signal circuit generation system and provides a relevant training dataset. The third describes a framework that uses an LLM to generate Verilog code for synthesizing convolutional neural network hardware accelerators. The fourth uses an LLM to refactor programs written in C with the goal of enabling high-level synthesis of hardware from C programs. The fifth describes an agent-based system that decomposes HDL code generation tasks into simpler sub-tasks to enable LLMs to produce correct code for larger-scale, more complex design problems. The sixth describes the use of an LLM to optimize hyperparameters for a standard cell placement system. Finally, the seventh uses retrieval-augmented generation (RAG) to improve RTL code generation.

The third section focuses on using **LLMs to improve electronic system correctness, reliability, and security**. Considerable engineering time is spent on validation and verification of designs, as well as other aspects of hardware-software system reliability and security. Improving the quality and speed of these processes would have a major impact on industry practice. The first paper in this section describes a method enabling LLMs to accept feedback from lower-level hardware synthesis tools and revise LLM-generated HDL code to correct design errors. The second describes an iterative RAG-based method of using LLMs to correct errors in Verilog HDL code. The third describes an LLM- and RAG-based HDL code debugging framework. The fourth describes an LLM-based method to identify functional and security defects in system-on-chip HDL specifications. Finally, the fifth describes using an LLM to identify patterns in, and track the temporal evolution of, hardware and Internet-of-Things vulnerabilities.

The fourth and final section focuses on using **LLMs to explain or model integrated circuits and systems**. Such work can provide tools that aid engineers in the design process so they can produce better designs in less time. The first paper describes using RAG to enable an LLM-based system to build knowledge graphs capturing dependencies and hierarchies in technical

documentation with the goal of supporting domain-specific technical queries. The second describes a method of using a multimodal LLM and RAG to reduce the human effort needed to determine the appropriate hardware device parameter values allowing good matches to device characteristic plots. The last paper describes using a transformer architecture neural network to predict basic block performance and power consumption directly from compiled software binaries.

We would like to thank those who submitted papers to the special issue for their trust and patience with the review process. We would also like to thank Editor in Chief Jiang Hu and Associate Managing Editor Clarissa Nemeth for their timely help and suggestions on managing the review process and preparing the special issue and Senior Associate Editor Jörg Henkel for his advice on defining and publicizing the special issue.

Received 25 June 2025; revised 25 June 2025; accepted 26 June 2025