Strengthening the Foundations of IC Physical Design and ML EDA Research

Vidya A. Chhabria
Arizona State Univ., USA
vachhabr@asu.edu

Vikram Gopalakrishnan
Arizona State Univ., USA
vgopal18@asu.edu

Andrew B. Kahng
UC San Diego, USA
abk@ucsd.edu

Sayak Kundu
UC San Diego, USA
sakundu@ucsd.edu

Zhiang Wang
UC San Diego, USA
zhw033@ucsd.edu

Bing-Yue Wu
Arizona State Univ., USA
bingyuew@asu.edu

Dooseok Yoon
UC San Diego, USA
d3yoon@ucsd.edu

ABSTRACT

Over the past year, IEEE CEDA DATC has continued to improve the DATC Robust Design Flow (RDF) while also advancing open infrastructure for research, including machine learning for electronic design automation (ML EDA). The 2024 RDF release includes new standalone and integrated global placement and macro placement engines, as well as a CCS-based delay calculator. Advances in baselines and benchmarks include the addition of new benchmarks for macro placement and logic gate sizing, as well as further efforts to establish calibrations of both optimizations and analyses to aid assessments of research progress in EDA. Additional efforts to promote open and reproducible research include refined proxy research enablements and enhanced ML EDA infrastructure through the development and use of new formats, the release of datasets, and the development of Python APIs in OpenROAD.

CCS CONCEPTS

• **Hardware → Physical design (EDA)**; **Methodologies for EDA**.

KEYWORDS

VLSI CAD, open-source, machine learning

ACM Reference Format:
Vidya A. Chhabria, Vikram Gopalakrishnan, Andrew B. Kahng, Sayak Kundu, Zhiang Wang, Bing-Yue Wu, and Dooseok Yoon. 2024. Strengthening the Foundations of IC Physical Design and ML EDA Research. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '24), October 27–31, 2024, New York, NY, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3676536.3697136

1 INTRODUCTION

The Design Automation Technical Committee (DATC) within the IEEE Council on EDA (CEDA) [45] seeks to address critical issues, needs, and community strategies in design automation. Since 2016, DATC has overseen the development of the Robust Design Flow (RDF), a comprehensive academic reference flow from RTL to GDSII, integrating multiple award-winning point tools alongside the OpenROAD [2] toolchain. RDF was created with two primary aims: (i) to preserve and integrate cutting-edge academic research tools, and (ii) to stimulate research in design flow optimization and cross-stage integration. A series of invited papers has chronicled updates to RDF [5–7, 13–15, 17–19] and highlighted DATC's evolving strategic priorities and development efforts.

In 2020 [6], with the development of OpenROAD, RDF integrated the OpenROAD app [56], creating a single open-source tool-based RTL-to-GDSII implementation flow, alongside a cross-stage multiple tool-based flow. The scope and mission of RDF were also updated, bringing attention to analysis and verification research; validation of research in a full-flow context; and infrastructure (from obfuscation and anonymization to metrics collection) to support ML-enabled EDA (ML EDA) research. RDF is currently built upon many academic tools, as shown in Table 1. This year, RDF includes the updates shown in bold. In this paper and our GitHub repository [43], we highlight improvements in the flow as well as new foundations for IC physical design and ML EDA research, including updates to baselines, benchmarks, ML EDA formats, and datasets. The main directions of DATC efforts in the past year include the following.

- **Recent improvements of RDF.** In its 2024 release, RDF has added a new dataflow-driven GPU-accelerated placement engine, DG-RePlAce, for global placement and macro placement, as well as the new multi-bit flip-flop clustering feature in OpenROAD and a composite current source (CCS) delay calculator in OpenSTA. These RDF improvements are available in open source through OpenROAD integration.
- **Advances in benchmarks and baselines.** This year, DATC efforts have added MemPool Cluster, a 10.5M-instance benchmark, as well as scaled versions (2x and 4x) of Ariane (in .v, .def format) and CT-Ariane [42] (in protobuf format) to the MacroPlacement repository [57]. Additionally, we have revisited the METRICS2.1 [16] work and seeded a "leaderboard" of calibration results for AES, IBEX, and JPEG designs in NanGate45 enablement, using OpenROAD-flow-scripts and obfuscated results from unnamed commercial synthesis and place-and-route (P&R) tools. These calibrations and initial leaderboard results are available in our RDF GitHub repository [43].
- **Foundations for ML EDA research.** Previous works such as [49] have emphasized the need for an open infrastructure for ML EDA. DATC efforts this year have included improved *proxies* and other key ML EDA enablements. Developments in 2024 feature benchmarks for logic gate sizing and a refined calibration method aimed

ICCAD '24, October 27–31, 2024, New York, NY, USA

ACM ISBN 979-8-4007-1077-3/24/10…$15.00
https://doi.org/10.1145/3676536.3697136

at bridging the gap between an open-source process design kit (PDK) and a commercial PDK. This year also saw efforts to make RDF more compatible with ML through Python APIs and support for NVIDIA's CircuitOps format. DATC activities have released a dataset for several benchmarks in this format. DATC also highlights initiatives such as the creation of artifact evaluation standards, applied to papers submitted to the MLCAD symposium [48], which are being adopted by other venues (e.g., LAD [46]) to encourage open and reproducible research.

Table 1: RDF-2024 Components.

Component	Tools
RTL generator	Chisel/FIRRTL
RTL obfuscation	ASSURE
Logic synthesis	Yosys, ABC
Hypergraph partitioning	SpecPart, TritonPart
DFT insertion	Fault
Floorplanning	TritonFP
Macro placement	TritonMP, RTL-MP, Hier-RTLMP, AutoDMP, **DG-RePlAce-AutoDMP**
Global placement	RePlAce, FZUplace, NTUPlace3, ComPLx, Capo, Eh?Placer, FastPlace3-GP, mPL5/6, DREAMPlace, **DG-RePlAce**
Detailed placement	OpenDP, MCHL, FastPlace3-DP, DPO
Flip-flop clustering	Mean-shift, FlopTray
Clock tree synthesis	TritonCTS
Global routing	FastRoute4-lefdef, NCTUgr, CUGR
Detailed routing	TritonRoute, NCTUdr, DrCU
Layout finishing	KLayout, Magic
Gate sizing	Resizer, TritonSizer
Parasitic extraction	OpenRCX
STA	OpenSTA, iTimerC
Database	OpenDB
Libraries/PDK	GF180MCU, NanGate45, SKY130, ASAP7, NCTUcell, ASAP5
Integrated app	OpenROAD
Benchmark conversion	RosettaStone
DTCO	PROBE3.0

In the following, Section 2 describes several recent advancements in RDF. Section 3 discusses recent improvements to benchmarks and baselines. Section 4 highlights recent advancements in ML EDA infrastructure for open and reproducible research. Section 5 describes a roadmap of planned DATC efforts over the next several years. We conclude in Section 6 with a summary of the past year's progress.

2 RECENT IMPROVEMENTS OF RDF

In this section, we highlight three main improvements made in RDF-2024: a dataflow-driven GPU-accelerated global placement framework, the new multi-bit flip-flop clustering feature in OpenROAD, and the calibration efforts in OpenSTA. Details of new Python APIs added to OpenROAD to support ML EDA are discussed in Section 4.2.

2.1 DG-RePlAce: Dataflow-Driven GPU-Accelerated Global Placement Framework

Global placement is a fundamental step in VLSI physical design which determines the locations of standard cells and macros in a layout. The advent of large machine learning accelerators with millions of standard cells, hundreds or even thousands of macros, and unique dataflow and datapath architectures has introduced significant challenges, particularly in terms of runtime and quality of results (QoR).

DG-RePlAce in RDF-2024 is a dataflow-driven GPU-accele-rated global placement engine integrated into OpenROAD. The flow of DG-RePlAce is shown in Figure 1, and is distinguished by the following key features. (i) In contrast to previous GPU-accelerated global placers such as DREAMPlace [24], it is fully integrated within the OpenROAD RTL-to-GDSII flow, and implemented in CUDA and C++ to eliminate dependencies on external frameworks such as PyTorch, which simplifies installation and deployment. (ii) It leverages both dataflow information and datapath regularity to guide global placement toward improved QoR, making it particularly effective for early architecture design space exploration. (iii) It is permissively open-sourced and designed for scalability, which supports adaptation to future problem instances.

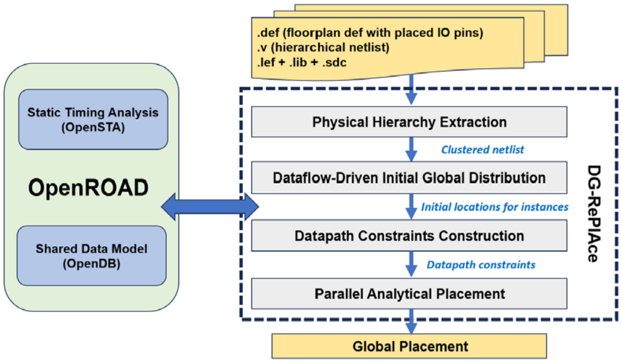

.def (floorplan def with placed IO pins)
.v (hierarchical netlist)
.lef + .lib + .sdc
Static Timing Analysis (OpenSTA)
OpenROAD
Shared Data Model (OpenDB)
Physical Hierarchy Extraction
Clustered netlist
Dataflow-Driven Initial Global Distribution
Initial locations for instances
Datapath Constraints Construction
Datapath constraints
Parallel Analytical Placement
DG-RePlAce
Global Placement

Figure 1: Overview of the DG-RePlAce flow.

We also improve the performance of DG-RePlAce using DG-RePlAce-AutoDMP, which autotunes the parameters of DG-RePlAce to boost performance, inspired by the approach used in AutoDMP [1]. Figure 2 shows the two main steps in DG-RePlAce-AutoDMP. (i) At left: the multi-objective optimization engine explores the placement space by tuning DG-RePlAce parameters. It evaluates placements using post-placement metrics such as RSMT-based wirelength, density, and RUDY-based congestion (see [1] for details on each metric). Along with the multi-objective Bayesian optimization algorithm (MOTPE) used by AutoDMP authors, we also support the powerful NSGA-II evolutionary algorithm [11]. We deploy NSGA-II via the Ray Tune framework [26]. (ii) At right: after sampling, the most promising candidates from the Pareto front are evaluated using post-route metrics such as routed wirelength, total negative slack (TNS), and power. These evaluations take place within a commercial EDA tool after timing optimization and routing. Figure 3 compares normalized metrics of post-route layouts achieved using various tools, including DG-RePlAce-AutoDMP, on the MemPool Group testcase [47] with a commercial 12nm enablement. For this example, we employ the NSGA-II optimization engine with 100 trials. We see that DG-RePlAce-AutoDMP improves all PPA metrics in comparison to DG-RePlAce.[1]

2.2 Multi-Bit Flip-Flop Clustering

Multi-bit flip-flops (MBFFs) are an important lever for power reduction. Using MBFFs instead of single-bit FFs decreases the effective number of clock sinks and total clock pin capacitance. As a result, wirelength and the total number of clock buffers required for clock distribution are both reduced, leading to clock power reduction. On the other hand, flip-flop movement during the MBFF clustering process can harm data path timing and increase combinational power.

[1] Permissively-licensed source code of DG-RePlAce-AutoDMP is available in [38].

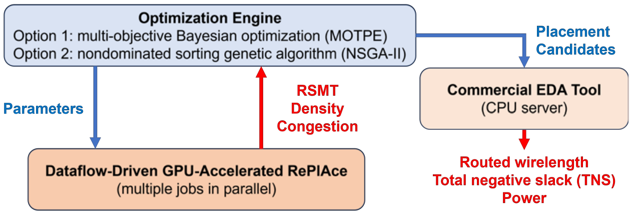
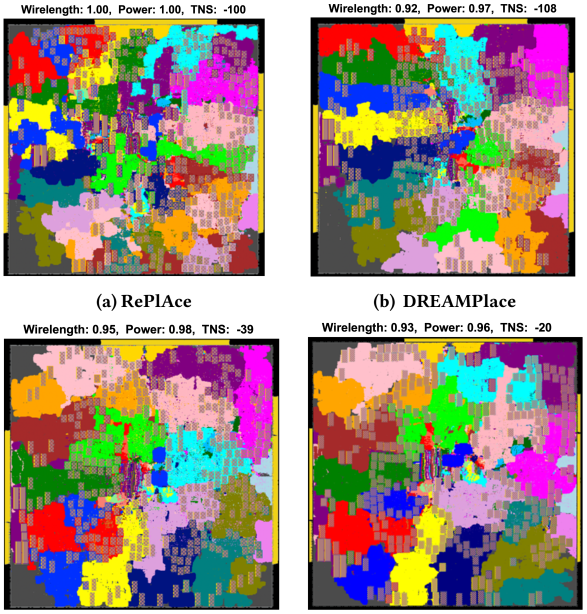

Optimization Engine
Option 1: multi-objective Bayesian optimization (MOTPE)
Option 2: nondominated sorting genetic algorithm (NSGA-II)
Placement Candidates
Parameters
RSMT
Density
Congestion
Commercial EDA Tool
(CPU server)
Routed wirelength
Total negative slack (TNS)
Power
Dataflow-Driven GPU-Accelerated RePlAce
(multiple jobs in parallel)

Figure 2: Overview of the DG-RePlAce-AutoDMP flow.


Wirelength: 1.00, Power: 1.00, TNS: -100
Wirelength: 0.92, Power: 0.97, TNS: -108
Wirelength: 0.95, Power: 0.98, TNS: -39
Wirelength: 0.93, Power: 0.96, TNS: -20

(a) RePlAce

(b) DREAMPlace

(c) DG-RePlAce

(d) DG-RePlAce-AutoDMP

Figure 3: Post-route layouts of MemPool Group [47] in a commercial 12nm enablement. To protect foundry IP, all metrics are normalized: wirelength and power are normalized relative to *RePlAce*, and total negative slack (TNS) is normalized relative to the clock period.

Work reported in [21] has enabled flip-flop clustering in OpenROAD. Additionally, 2-bit and 4-bit multi-height MBFF standard cells for the ASAP7 [10] enablement have been added to OpenROAD-flow-scripts [51]. Figure 4 compares placements and corresponding clock trees using (a) single-bit FFs and (b) MBFFs, for the JPEG [50] design in ASAP7. In the figure, the single-bit FFs are highlighted in red, and the MBFFs are highlighted in green. The implementation with MBFFs uses 181 clock buffers, while the implementation with single-bit FFs uses 245 clock buffers.

2.3 OpenSTA Calibrations

Dynamic Power Calibration. Last year's DATC efforts [18] highlighted a recently-added dynamic power analysis capability of OpenSTA [54]. Given an input *value change dump* (VCD) file, OpenSTA performs vectored dynamic power analysis. [18] showed differences in dynamic power reported by OpenSTA and an unnamed commercial tool, using a VCD file for an Ariane-133 [36] RISC-V core implemented in NanGate45 [40]. Improvements to dynamic power estimation were subsequently made; Figure 5 shows an updated calibration against the same unnamed commercial tool for the same design.

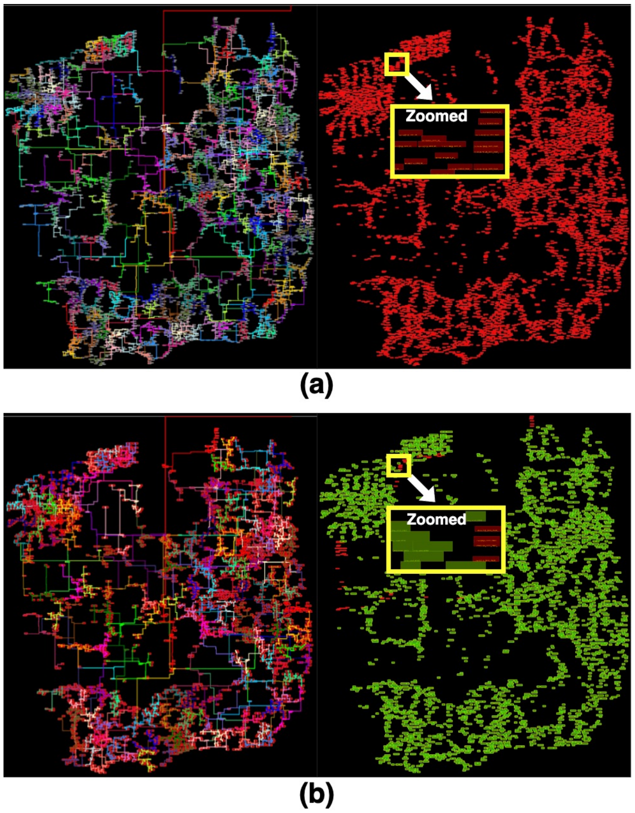


Figure 4: Visualization of the clock tree (left) and flip-flop placement (right) for the JPEG design in ASAP7: (a) using single-bit flip-flops only, and (b) with multi-bit flip-flops.

CCS Timing Calibration. The OpenSTA static timing analysis engine [54] now supports delay calculation with the CCS timing model. Figure 6(a) compares endpoint arrival times for the post-route JPEG design using the corresponding ASAP7 CCS libraries [10]. The synthesis and place-and-route are done using OpenROAD-flow-scripts (ORFS). For each timing endpoint, the start-end timing path with the worst slack, as identified by OpenSTA, is analyzed by both OpenSTA and an unnamed commercial timer. The error histogram in Figure 6(b) shows the distribution of deviations between the two timers. For these examples, all Verilog, DEF, SPEF and SDC files, along with the 5-worst JSON, endpoints JSON, and timing report viewer, are open-sourced in [44]. As described in [6], the 5-worst JSON format contains detailed information for the top-5 worst timing paths, block-level worst negative slack (WNS), total negative slack (TNS), and number of failing endpoints (FEP). The endpoint JSON format includes setup slack values at every flip-flop D pin.

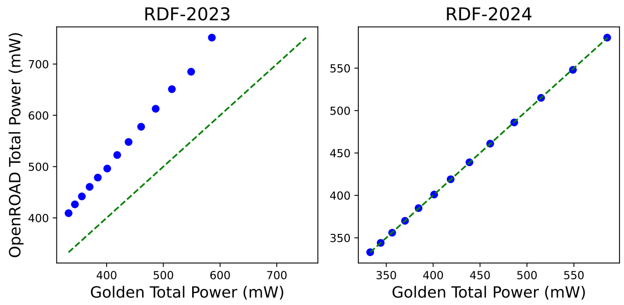


Figure 5: Vectored dynamic power correlation between OpenSTA and an unnamed commercial tool (Ariane design in NanGate45 with bsg_fakeram SRAM models). Left: RDF-2023. Right: RDF-2024.

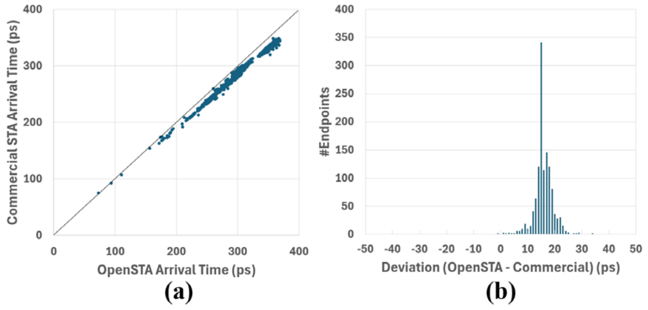


Figure 6: Correlation analysis results on JPEG with ASAP7 CCS libraries: (a) endpoint arrival time comparison between OpenSTA and commercial STA results, and (b) distribution of deviations from (a).

3 BENCHMARKING AND BASELINES

In this section, we present updated macro placement benchmarks and recent flow tuning efforts.

3.1 Updated Benchmarks for Macro Placement

RDF-2024 brings several additional public benchmarks for macro placement. The MemPool Cluster [4] testcase in open enablement augments MacroPlacement effort [8] [57], which was introduced in RDF-2022. MemPool Cluster has 1,296 macros and approximately 10.5M instances in the NanGate45 enablement. Furthermore, the well-studied Ariane-133 testcase [36], which has 133 macros and approximately 117K instances in the NanGate45 enablement, has been scaled into ×2 and ×4 derivatives as described for "quantified suboptimality" studies in [12]. Figures 7(a) and (b) show the macro placements for Ariane-133×2 and Ariane-133×4, respectively. Macro placements on the left of the figure are generated by Hier-RTLMP [22], while macro placements on the right are produced by DG-RePlAce [38]. As with previous open benchmarks, TCL flow scripts are publicly available in the *MacroPlacement* repository [57], to facilitate synthesis using Cadence Genus v21.1 and post-P&R evaluation of macro placement solutions using Cadence Innovus v21.1. We also note that Google Brain's Circuit Training for macro placement [25] [41] published another 133-macro version of Ariane design, in protobuf format with cell names corresponding to a TSMC 7nm (240nm cell height) library [42]. To provide additional testcases that reflect macro placement in sub-10nm technology, we have created CT-Ariane-133×2 and CT-Ariane-133×4 derivatives (in protobuf format), which are also available in the *MacroPlacement* repository [57].

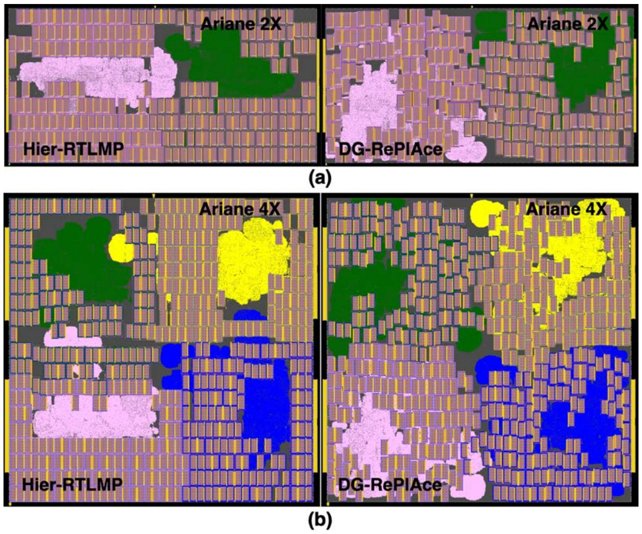


Figure 7: Macro placement solutions generated using Hier-RTLMP (left) and DG-RePlAce (right) for (a) Ariane-133×2 and (b) Ariane-133×4 in NanGate45.

3.2 Flow Tuning for Calibration and Progress

The work in [16] presented an open-source script to tune hyperparameters for OpenROAD-flow-scripts (ORFS) using Ray Tune [26] and demonstrated significant improvements in power, performance and area (PPA) for different target designs. Over the past several years, OpenROAD has introduced new features and added new flow knobs (e.g., detailed placement optimizer, resizer, etc.) that enable better PPA outcomes. We have revisited the study of [16], with autotuning of the hyperparameters listed in Table 2. In the table, newly added hyperparameters compared to [16] are highlighted in bold. In each autotuning run, the search algorithm Optuna [3] is used to generate 1,000 trials, with 20 at a time running in parallel, aimed at optimizing either performance or area. In our runs, the sweeping range for *CLOCK_PERIOD* is 0.1ns to 10ns for performance optimization, and 3ns to 10ns for area optimization.

The results in [16] do not shed any light on the strength of commercial baselines. To address this gap, we also perform hyperparameter tuning of commercial synthesis and place-and-route (P&R) tools. We create a leaderboard (Table 3) for the AES [50], IBEX [50], and JPEG designs using the NanGate45 [40] enablement, where the goal is to improve performance or area metrics. Table 3 provides three baselines: (a) **Comm**: using unnamed commercial synthesis and P&R tools, (b) **OR**: using OpenROAD-flow-scripts (ORFS), and (c) **OR***: using the best synthesized netlist from Comm with P&R executed through ORFS. In the OR* flow, the best Comm synthesis netlist optimized for area is used for area-focused runs, while the best Comm synthesis netlist optimized for performance is used for performance-focused runs. To ensure consistent evaluation, we use OpenRCX [53] for SPEF extraction and OpenSTA [54] for power and performance reporting. In Table 3, the effective clock period (EffCP) is calculated

Table 2: Tunable Tool and Design Parameters. The sweeping range for *CLOCK_PERIOD* is 0.1ns to 10ns for performance optimization and 3ns to 10ns for area optimization.

Parameters	Description	Type	Range
CLOCK_PERIOD	Target clock period (ns)	float	–
CORE_UTIL	Target core utilization (%)	int	[20, 99]
GP_PAD	Cell padding for global placement (site)	int	[0, 4]
DP_PAD	Cell padding for detailed placement (site)	int	[0, 4]
ENABLE_DPO	Detailed placement optimization	int	[0, 1]
PIN_LAYER_ADJUST	Layer resource adjustment during global routing (%) for metal2 and metal3 layers	float	[0.2, 0.7]
ABOVE_LAYER_ADJUST	Layer resource adjustment during global routing (%) for metal4 and above layers	float	[0.2, 0.7]
PLACE_DENSITY_LB_ADDON	Additional lower bound increase of the target local global placement density (%)	float	[0.00, 0.99]
FLATTEN	Design hierarchy flattening	int	[0, 1]
PINS_DISTANCE	Minimum IO pin distance (#tracks)	int	[1, 3]
CTS_CLUSTER_SIZE	Target CTS sink cluster size	int	[10, 40]
CTS_CLUSTER_DIAMETER	Target CTS sink cluster diameter (μm)	int	[80, 120]
TNS_END_PERCENT	Percentage of violating endpoints to repair	int	[0, 100]

Table 3: Calibration ("leaderboard") of performance and area outcomes achievable by Comm: (unnamed) commercial synthesis and P&R tools, OR: OpenROAD-flow-scripts [51] (commit: 3b4c59a), and OR*: the best synthesis netlist from Comm with P&R using OpenROAD-flow-scripts. Diff (%) indicates the percentage difference from outcomes of the best commercial result.

Case	Goal	Tool	EffCP (ns)	Diff (%)	Area (μm^2)	Diff (%)	Power (mW)
AES	Perf	Comm	**0.582**	**NA**	40679	NA	349.8
		OR	**0.915**	**57.1**	65861	61.9	396.9
		OR*	**0.694**	**19.1**	58995	45.0	236.1
	Area	Comm	1.285	NA	**14166**	**NA**	85.1
		OR	1.098	-14.55	**27433**	**93.7**	326.8
		OR*	1.628	26.7	**20512**	**44.8**	90.8
IBEX	Perf	Comm	**1.123**	**NA**	68843	NA	794.9
		OR	**2.313**	**106.0**	44825	-34.9	92.0
		OR*	**1.654**	**47.3**	47969	-30.3	473.9
	Area	Comm	4.273	NA	**22331**	**NA**	39.3
		OR	4.346	1.7	**37699**	**68.8**	50.4
		OR*	4.134	-3.3	**28077**	**25.7**	56.6
JPEG	Perf	Comm	**0.616**	**NA**	156700	NA	752.2
		OR	**1.360**	**120.7**	310723	98.3	970.1
		OR*	**0.899**	**45.9**	182624	16.5	441.8
	Area	Comm	14.725	NA	**59521**	**NA**	20.3
		OR	1.602	-89.1	**117217**	**96.9**	811.1
		OR*	2.138	-85.5	**68694**	**15.4**	132.8

as the target clock period minus the worst setup slack. The reported area refers to the core area of the design, and power represents the total power of the design computed at the effective clock period.

OpenROAD is evolving rapidly, and maintaining a leaderboard will allow us to understand its limitations and track its progress [27–29]. Our new leaderboard also enables proposed EDA algorithm improvements to be easily assessed using standardized evaluators for PPA metrics, and in the OpenROAD full-flow context. To support this purpose, we have uploaded our autotuning scripts, evaluation flow and baseline solutions to the RDF GitHub repository [43]. For anonymity, all design hierarchies are flattened, and all instances, nets, and generated vias are renamed to generic identifiers (e.g., instances as `i1`, `i2`, ...; nets as `n1`, `n2`, ...; and generated vias as `via1`, `via2`, ...). In addition, all tool-specific headers and keywords are removed from the Verilog, DEF, and SDC files.

4 ML DATA AND RESEARCH ENABLEMENTS

We now describe progress made in ML EDA data and research enablement, including proxy design enablement, data formats and datasets for ML EDA, and ML gate sizing benchmarks. We also highlight the use of LLMs for physical design, and efforts toward reproducible ML EDA research.

4.1 Proxy Design Enablement: BEOL Parameters and Multi-PVT Tuning

Calibration of proxy design enablement was introduced in [18], which used [26] to autotune scaling factors for standard-cell timing and power models. This work narrowed the gap between the open-source ASAP7 enablement and an unnamed leading-edge commercial 7nm technology node by scaling internal power, switching power, and cell delay in Liberty files. In RDF-2024, we add setup/hold timing and pin capacitance, as well as BEOL resistance and capacitance, to the set of autotuned scaling factors.

Figure 8 shows the updated autotuning flow and impact of the added BEOL RC scaling factors. Synthesis and P&R are performed using Cadence Genus v21.1 and Innovus v21.1, respectively. Loss is the average of Mean Absolute Percentage Error (MAPE) of total power and MAPE of effective clock period, across all target clock periods. While the addition of BEOL RC scaling factors reduces loss by only 0.33%, it enables autotuning solutions with less-extreme standard-cell scaling factors.

We confirm the robustness of our autotuning-based proxy enablement at multiple PVT (Process, Voltage, Temperature) targets. That is, we vary process from FF (fast NMOS and fast PMOS) to SS (slow NMOS and slow PMOS); voltage from UHV (ultra-high voltage) to ULV (ultra-low voltage); and temperature from HT (high temperature) to LT (low temperature). To protect the foundry's IP, specific numbers are not disclosed. Figure 9 shows power-performance outcomes for JPEG implementation across a range of target clock periods (CP), for autotuned enablements. Part (a) of the figure shows combinations of fast process, high voltage and high temperature. Part (b) shows combinations of slow process, low voltage and low temperature. The loss values for all four PVT corners are below 10%, with three of them under 2%. We have also performed autotuning runs for the more traditional best corners (FF-(U)HV-LT) and worst corners (SS-(U)LV-HT), obtaining loss values under 4.6% for the best corners and under 2.5% for the worst corners. Our results suggest that the open-source PDK at a single PVT corner (i.e., ASAP7's (TT, 0.7V, 25C) corner in our studies) can be effectively tuned to cover a wide range of PVT corners in a closed-source PDK. We have open-sourced the autotuning scripts for proxy enablement in the RDF GitHub repository [43].

4.2 Data Formats and Datasets for ML EDA

A year ago, RDF-2023 [18] presented a roadmap of Python APIs in OpenROAD, with an end goal of OpenROAD becoming a *playground* for EDA researchers where ML algorithms can be easily integrated into EDA tools. Since then, there have been several advancements toward this goal. (1) CircuitOps [23], a standard data format to store design data in an AI-friendly way, was developed. In CircuitOps, the design data is represented as Labeled Property Graphs (LPGs) backed by intermediate representation tables (IR Tables), simplifying the process of custom dataset generation for ML applications. (2) Numerous Python APIs in OpenROAD have been developed, offering faster dataset generation when compared to TCL interfaces [9]. The newly-developed APIs include reading design files, querying timing properties from OpenROAD (e.g., slacks, load capacitances, and arrival times of pins in the design), and back-annotating properties into OpenROAD. These APIs not only enable the direct training of ML models using design data within OpenROAD, but also establish a feedback loop from ML algorithms to design, feeding ML inference results back into OpenROAD by modifying the OpenROAD database through the APIs. This infrastructure using CircuitOps is shown in Figure 10. A detailed overview of all newly developed APIs is available at the ASP-DAC24-Tutorial GitHub repository [34]. (3) Using these Python APIs, as a part of DATC activities, we have created IR tables using data from post-route DEF for eight designs in NanGate45 [40], five designs in SKY130HD [55], and four designs in ASAP7 [10]; all of these designs are available in the OpenROAD-flow-scripts repository. The post-route DEF data is produced by running the OpenROAD-flow-scripts default flow, and the IR tables are made available to boost research on ML EDA applications and seed further data generation by the community. Details of these designs and the IR tables are available at the CircuitOps GitHub repository [58].

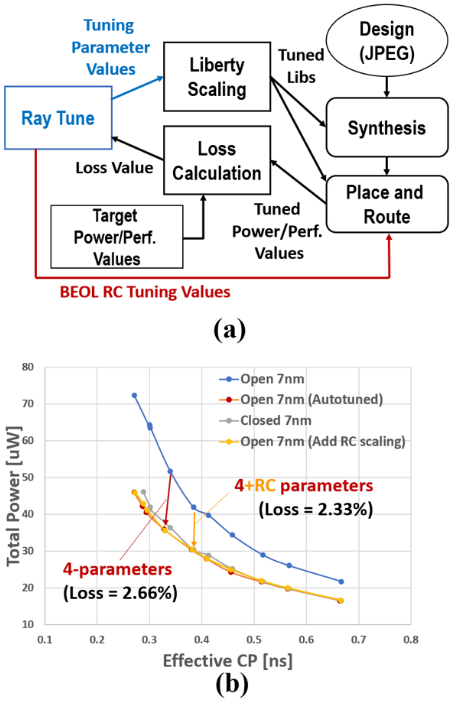


Figure 8: Addition of BEOL RC parameter tuning: (a) autotuning flow and (b) results of power and performance autotuning. “4-parameters” indicates tuning of four standard-cell scaling factors (delay, power, setup/hold timing, and pin capacitance). “4+RC parameters” indicates the addition of BEOL resistance and capacitance scaling factors.

4.3 ML Gate Sizing Benchmarks

The 2024 ICCAD Contest Problem C on gate sizing [31] spurs development of new algorithms to minimize leakage power while meeting timing and electrical constraints. The contest objective is a function of runtime, leakage power, total negative timing slack, and slew and load violations. The contest benchmarks [33] include eight designs from the MacroPlacement GitHub repository [57], along with two designs from OpenCores [50], all implemented in ASAP7. These benchmarks are provided in two formats: (1) standard EDA files, which consist of SDC, LIB, LEF, DEF and gate-level netlist Verilog files, and (2) NVIDIA’s CircuitOps [23] IR Table format. The benchmarks are created from post-placement solutions produced by an unnamed

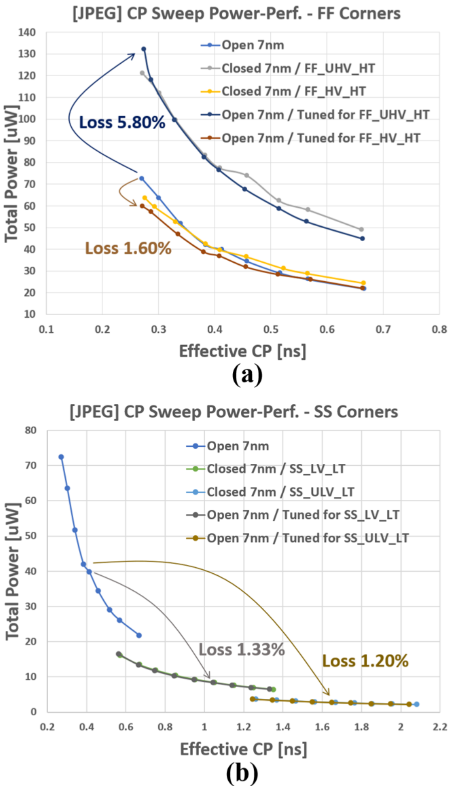


Figure 9: Power-Performance of ASAP7 autotuned at multiple PVT corners, relative to an unnamed commercial 7nm technology: (a) FF corners and (b) SS corners.

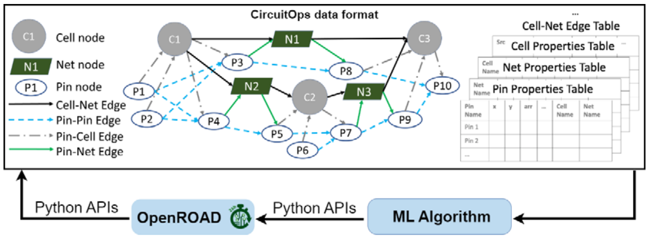


Figure 10: Integration of ML algorithms with OpenROAD Python APIs and CircuitOps.

commercial tool. The contest also provides example scripts that leverage OpenROAD's Python APIs, to illustrate the use of OpenROAD and its integration with ML approaches. The contest benchmarks, along with supporting scripts and examples, are available at [33].

4.4 Large Language Models for Physical Design

The EDA Corpus dataset [32] [39] is the first open-source dataset designed to support large language model (LLM) based research (e.g., chatbot development and script generation) in physical design. EDA Corpus includes 943 OpenROAD prompt-script data pairs, comprising 373 pairs of prompts and OpenROAD Python scripts corresponding to various stages of the physical design flow, along with 570 pairs of prompts and OpenROAD Python scripts to query information from the OpenROAD database. The dataset also 590 question-answer pairs, including 181 pairs related to general OpenROAD modules, 190 pairs related to the OpenROAD tool user manual, and 219 pairs related to the OpenROAD flow. All data pairs in EDA Corpus have been reviewed by OpenROAD experts and are formatted as prompt-script or question-answer pairs, making them easy to integrate into LLM training workflows.

The OpenROAD-Assistant project [30] [52] is an open-source LLM built on the EDA Corpus dataset. It is designed to help OpenROAD users learn the tool, and/or to reduce engineering efforts for physical designers who use OpenROAD. OpenROAD-Assistant answers questions related to OpenROAD and can generate Python scripts to perform physical design tasks or to query information from the OpenROAD database, with natural language prompts.

4.5 Toward Reproducible ML EDA Research

Artifact evaluation is a critical initiative to promote open, reproducible research within the EDA/CAD community. The process allows authors to submit the codes, datasets, training scripts, and inference scripts used to produce the key results of their accepted papers. These submitted elements, known as artifacts, undergo peer evaluation to check if they meet standards of availability (*available* badge), functionality (*functional* or *reviewed* badge), and reproducibility (*reproducible* badge) defined by ACM/IEEE. Papers that meet these standards are awarded badges. In 2024, the MLCAD symposium adapted ACM badges [35] and developed artifact evaluation standards based on MLCommons [37] to create standards tailored to the ML EDA community. These standards address challenges such as the use of proprietary tools and PDKs, as documented in [48]. The DATC sees this process as a model for other conferences, encouraging a broader adoption of artifact evaluation criteria across the EDA/CAD research landscape. DATC activities in the next year will refine these standards by working with IEEE CEDA based on community inputs and past experiences.

5 IEEE CEDA DATC ROADMAP

The mission of DATC is to serve as a central organization and platform that addresses key challenges in design automation, facilitates collaboration on public design flows and testcases, and organizes relevant workshops, meetings, and publications. In recent years, RDF efforts have focused on building public design flows, defining key metrics, and enabling ML EDA. The overarching goal is to ensure continuous progress in EDA research by maintaining baseline calibrations, tracking metrics, and integrating point CAD tools into a cohesive flow to promote cross-stage research and innovation, as highlighted in [20].

Challenges. Despite notable progress, several challenges must be addressed as DATC continues its efforts to accelerate innovation and the advancement of EDA research. In reviewing the history of past DATC RDF efforts, the following challenges are apparent.

- *Limited external contributions to DATC activities.* External contributions to DATC RDF have been limited, mainly due to a lack of community awareness, established contribution guidelines, and incentives for maintaining open-source repositories. The absence of diverse academic contributions also limits the definition of best results possible on specific benchmarks, impeding research progress.
- *Fragmented repositories and tools.* Over the past years, DATC has annually released new repositories to showcase its activities. However, this approach has proven ineffective, as maintaining multiple fragmented repositories complicates updates and weakens the impact of the research enablement efforts.
- *Reproducibility of research.* EDA research continues to face challenges of reproducibility due to issues related to proprietary designs, PDKs, and commercial EDA tools. Furthermore, varying flows with different parameters, along with the computational demands of ML-driven EDA, exacerbate these reproducibility challenges. The closed nature of certain research practices further limits open-source contributions and collaboration.

Short-term activities. DATC will take the following short-term actions to address these challenges:

- *Unified RDF GitHub repository with continuous integration (CI).* Beginning this year, DATC will maintain a single, well-organized GitHub repository to facilitate community contributions and ensure long-term sustainability. This repository will provide clear documentation and contribution standards, and will feature two flows: one integrating OpenROAD tools, and the other based on individual tools (and file-based IO) via precompiled binaries or submodules. A CI system will be implemented to streamline contributions and maintain flow quality. Annual releases will document baselines, benchmarks, and ML EDA enablements in this central repository.
- *Incentives for external contributions.* DATC will incentivize external contributions by offering developer certificates and certificates of recognition, which we plan to award annually at ICCAD during the ICCAD Contest Special Session. This approach will encourage greater participation from diverse contributors across the research community.
- *Support for open-source research artifacts.* DATC will lead the artifact evaluation processes at key conferences such as LAD and MLCAD. Expanding these efforts to other venues, DATC will establish reproducibility challenges and award badges based on EDA-specific IEEE and ACM artifact evaluation policies, thus promoting open and reproducible research.

Long-term activities. In the long term, DATC will focus on:

- *Expanding community awareness and engagement.* DATC will increase its presence through presentations, invited talks, and outreach. This will help raise awareness of DATC RDF and encourage contributions from a broader audience.
- *Continuous improvement of testcases and tools.* By enhancing testcases integrated with the CI system, DATC will strengthen the

RDF while also providing benchmarks and baselines for academic research.
- *Refining artifact evaluation standards.* As the artifact evaluation process matures, DATC will engage with key stakeholders to enhance policies for badge allocation in EDA research. This will include organizing peer-reviewed reproducibility challenges to further incentivize open and reproducible research practices.

Metrics of success. To measure the success and impact of these activities, DATC will track several key metrics, such as the number of external contributions to the RDF GitHub repository, the number of developer certificates and contributions recognized annually at ICCAD, the number of reproducibility badges awarded at EDA conferences, the number of new point tools incorporated into the RDF flow, and the number of testcases and designs integrated into the CI system. These metrics will be compiled and reported annually (e.g., at ICCAD) to ensure transparency and continuous improvement.

6 CONCLUSIONS

In this paper, we have described several key DATC RDF efforts made over the past year.

Recent improvements of RDF include a new dataflow-driven GPU-accelerated global placement framework (DG-RePlAce), the new multi-bit flip-flop clustering feature in OpenROAD, and calibration of OpenSTA. DG-RePlAce creates an effective global placement by leveraging dataflow information and regularity. The multi-bit flip-flip clustering feature is an important lever for power reduction. For OpenSTA calibrations, VCD-based dynamic power correlation has improved since last year's RDF, and CCS timing calibration data has been added.

Advances in benchmarks and baselines include updated benchmarks for macro placement and flow tuning for calibration and progress tracking. A new design, MemPool Cluster (1,296 macros and 10.5M instances), as well as 2x and 4x scaled versions of Ariane (in NanGate45) and CT-Ariane (7nm design in protobuf format), have been uploaded to the *MacroPlacement* repository. Furthermore, to calibrate and track the progress of OpenROAD, a baseline leaderboard for performance and area has been established for the AES, IBEX, and JPEG designs by autotuning the flow parameters of ORFS and an unnamed commercial synthesis and P&R tools. Obfuscated commercial tool results, along with the ORFS designs and evaluation scripts, are uploaded to the RDF GitHub repository for reference.

Foundations for ML EDA research have been strengthened on multiple fronts. Adding setup/hold constraints, pin capacitances and BEOL RC values to the autotuning of proxy design enablements leads to less-extreme scaling factors and robust matching of power-performance across PVT corners. For an AI-friendly way of storing design data, CircuitOps provides a standard data format that simplifies custom data generation for ML applications. Additionally, numerous Python APIs in OpenROAD enable fast data generation for ML as well as seamless integration of ML inference results back into OpenROAD. For LLM research in physical design, the open-source EDA Corpus dataset provides numerous, well-curated prompt-script and question-answer data pairs which can be easily integrated into LLM training workflows. EDA Corpus serves as the foundation of OpenROAD-Assistant, the first open-source LLM designed to assist OpenROAD users.

Last, the *IEEE CEDA DATC Roadmap* will frame and inform future efforts to provide a central platform that addresses key challenges in design automation. In recent years, RDF has focused on building public design flows, defining metrics and calibrations, and enabling ML-driven EDA. However, the accumulation of tools, codes and other research infrastructure in RDF has exposed other problems, such as lack of external contributions, fragmented repositories, and limited progress toward reproducibility in EDA research. To address these issues, DATC will unify its GitHub repository, incentivize external contributions with certificates, and lead open-source artifact evaluation efforts. Long-term goals include expanding community engagement, improving test cases and tools, and refining reproducibility standards, with success metrics such as contributions and reproducibility badges.

7 ACKNOWLEDGMENTS

We thank Rongjian Liang from NVIDIA for supporting CircuitOps infrastructure and the ML EDA dataset generation. We also thank Matt Liberty from Precision Innovations, Inc. for his help with addressing issues in OpenROAD and OpenSTA. This work is supported in part by the IEEE Council on Electronic Design Automation.

REFERENCES

[1] A. Agnesina, P. Rajvanshi, T. Yang, G. Pradipta, A. Jiao, B. Keller, B. Khailany and H. Ren, "AutoDMP: Automated DREAMPlace-Based Macro Placement", *Proc. ISPD*, 2023, pp. 149-157.

[2] T. Ajayi, V. A. Chhabria, M. Fogaça, S. Hashemi, A. Hosny et al., "Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project", *Proc. DAC*, 2019, pp. 76:1-4.

[3] T. Akiba, S. Sano, T. Yanase, T. Ohta and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework", *Proc. SIGKDD*, 2019, pp. 2623-2631.

[4] M. Cavalcante, S. Riedel, A. Pullini and L. Benini, "MemPool: A Shared-L1 Memory Many-Core Cluster with a Low-Latency Interconnect", *Proc. DATE*, 2021, pp. 701-706.

[5] J. Chen, I. H.-R. Jiang, J. Jung, A. B. Kahng, V. N. Kravets, Y.-L. Li, S.-T. Lin and M. Woo, "DATC RDF-2019: Towards a Complete Academic Reference Design Flow", *Proc. ICCAD*, 2019, pp. 1-6.

[6] J. Chen, I. H.-R. Jiang, J. Jung, A. B. Kahng, V. N. Kravets, Y.-L. Li, S.-T. Lin and M. Woo, "DATC RDF-2020: Strengthening the Foundation for Academic Research in IC Physical Design", *Proc. ICCAD*, 2020, pp. 1-6.

[7] J. Chen, I. H.-R. Jiang, J. Jung, A. B. Kahng, S. Kim, V. N. Kravets, Y.-L. Li, R. Varadarajan and M. Woo, "DATC RDF-2021: Design Flow and Beyond", *Proc. ICCAD*, 2021, pp. 1-6.

[8] C.-K. Cheng, A. B. Kahng, S. Kundu, Y. Wang and Z. Wang, "Assessment of Reinforcement Learning for Macro Placement", *Proc. ISPD*, 2023, pp. 158-166.

[9] V. A. Chhabria, W. Jiang, A. B. Kahng, R. Liang, H. Ren, S. S. Sapatnekar and B.-Y. Wu, "OpenROAD and CircuitOps: Infrastructure for ML EDA Research and Education", *Proc. VTS*, 2024, pp. 1-4.

[10] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy and G. Yeric, "ASAP: A 7-nm FinFET Predictive Process Design Kit", *Microelectronics Journal* 53 (2016), pp. 105-115.

[11] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", *Trans. on Evolutionary Computation* 6(2) (2002), pp. 182-197.

[12] L. Hagen, J. H. Huang and A. B. Kahng, "Quantified Suboptimality of VLSI Layout Heuristics", *Proc. DAC*, 1995, pp. 216-221.

[13] J. Jung, I. H.-R. Jiang, J. Chen, S.-T. Lin, Y.-L. Li and V. N. Kravets, "DATC RDF: An Academic Flow from Logic Synthesis to Detailed Routing", *Proc. ICCAD*, 2018, pp. 1-4.

[14] J. Jung, I. H.-R. Jiang, J. Chen, S.-T. Lin, Y.-L. Li, V. N. Kravets and G.-J. Nam, "DATC RDF: An Open Design Flow from Logic Synthesis to Detailed Routing", *Proc. Workshop on Open-Source EDA Technology*, 2018, pp. 1-4.

[15] J. Jung, I. H.-R. Jiang, G.-J. Nam, V. N. Kravets, L. Behjat and Y.-L. Li, "OpenDesign Flow Database: The Infrastructure for VLSI Design and Design Automation Research", *Proc. ICCAD*, 2016, pp. 42:1-6.

[16] J. Jung, A. B. Kahng, S. Kim and R. Varadarajan, "METRICS2.1 and Flow Tuning in the IEEE CEDA Robust Design Flow and OpenROAD", *Proc. ICCAD*, 2021, pp. 1-9.

[17] J. Jung, A. B. Kahng, R. Varadarajan and Z. Wang, "IEEE CEDA DATC: Expanding Research Foundations for IC Physical Design and ML-Enabled EDA", *Proc. ICCAD*,

2022, pp. 1-8.

[18] J. Jung, A. B. Kahng, S. Kundu, Z. Wang and D. Yoon, “IEEE CEDA DATC Emerging Foundations in IC Physical Design and MLCAD Research”, *Proc. ICCAD*, 2023, pp. 1-8.

[19] J. Jung, P.-Y. Lee, Y. Wu, N. K. Darav, I. H. Jiang, V. N. Kravets, I. H.-R. Jiang, and V. N. Kravets, “DATC RDF: Robust Design Flow Database”, *Proc. ICCAD*, 2017, pp. 872-873.

[20] A. B. Kahng, “Looking Into the Mirror of Open Source”, *Proc. ICCAD*, 2019, pp. 1-8.

[21] A. B. Kahng, S. Kundu and S. Thumathy, “Scalable Flip-Flop Clustering Using Divide and Conquer For Capacitated K-Means”, *Proc. GLSVLSI*, 2024, pp. 177-184.

[22] A. B. Kahng, R. Varadarajan and Z. Wang, "Hier-RTLMP: A Hierarchical Automatic Macro Placer for Large-scale Complex IP Blocks", *Trans. on CAD* 42(5) (2023), pp. 1552-1565.

[23] R. Liang, A. Agnesina, G. Pradipta, V. A. Chhabria and H. Ren, “Invited Paper: CircuitOps: An ML Infrastructure Enabling Generative AI for VLSI Circuit Optimization”, *Proc. ICCAD*, 2023, pp. 1-6.

[24] P. Liao, S. Liu, Z. Chen, W. Lv, Y. Lin and B. Yu, “DREAMPlace 4.0: Timing-Driven Global Placement with Momentum-Based Net Weighting”, *Proc. DATE*, 2022, pp. 939-944.

[25] A. Mirhoseini, A. Goldie, M. Yazgan et al., “A Graph Placement Methodology for Fast Chip Design”, *Nature* 594 (2021), pp. 207-212.

[26] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez and I. Stoica, “Tune: A Research Platform for Distributed Model Selection and Training”, *arXiv:1807.05118*, 2018. https://arxiv.org/abs/1807.05118

[27] I. M. Piatak, V. A. Antropov, O. T. De Laubenque and V. A. Yurchenko, “Open-Source and Non-Commercial Software for Digital ASIC Design”, *IEEE Intl. Conf. on Electrical Engineering and Photonics*, 2023, pp. 91-94.

[28] P. Sauter, T. Benz, P. Scheffler, F. K. Gurkaynak and L. Benini, “Insights from Basilisk: Are Open-Source EDA Tools Ready for a Multi-Million-Gate, Linux-Booting RV64 SoC Design?”, *arXiv:2405.04257*, 2024. https://arxiv.org/abs/2405.04257

[29] P. Sauter, T. Benz, P. Scheffler, Z. Jiang, B. Muheim, F. K. Gurkaynak and L. Benini, “Basilisk: Achieving Competitive Performance with Open EDA Tools on an Open-Source Linux-Capable RISC-V SoC”, *arXiv:2405.03523*, 2024. https://arxiv.org/abs/2405.03523

[30] U. Sharma, B.-Y. Wu, S. R. D. Kankipati, V. A. Chhabria and A. Rovinski, “OpenROAD-Assistant: An Open-Source Large Language Model for Physical Design Tasks”, *Proc. MLCAD*, 2024, pp. 1-7.

[31] B.-Y. Wu, R. Liang, G. Pradipta, A. Agnesina, H. Ren and V. A. Chhabria, “2024 ICCAD CAD Contest Problem C: Scalable Logic Gate Sizing Using ML Techniques and GPU Acceleration”, *Proc. ICCAD*, 2024.

[32] B.-Y. Wu, U. Sharma, S. R. D. Kankipati, A. Yadav, B. K. George, S. R. Guntupalli, A. Rovinski and V. A. Chhabria, “EDA Corpus: A Large Language Model Dataset for Enhanced Interaction with OpenROAD”, *Proc. LAD*, 2024.

[33] 2024_ICCAD_Contest_Gate_Sizing_Benchmark. https://github.com/ASU-VDA-Lab/2024_ICCAD_Contest_Gate_Sizing_Benchmark

[34] ASP-DAC24-Tutorial. https://github.com/ASU-VDA-Lab/ASP-DAC24-Tutorial

[35] ACM Artifact Review and Badging. https://www.acm.org/publications/policies/artifact-review-and-badging-current

[36] Ariane RISC-V CPU Repo. https://github.com/openhwgroup/cva6

[37] Artifact evaluation.
https://github.com/ctuning/artifact-evaluation/blob/master/docs/reviewing.md

[38] DG-RePlAce-AutoDMP.
https://github.com/ABKGroup/DG-RePlAce-AutoDMP/tree/main

[39] EDA-Corpus. https://github.com/OpenROAD-Assistant/EDA-Corpus

[40] FreePDK45. https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts/tree/master/flow/platforms/nangate45

[41] Google Brain Circuit Training. https://github.com/google-research/circuit_training/

[42] Google Brain Ariane testcase (protobuf). https://github.com/google-research/circuit_training/tree/main/circuit_training/environment/test_data/ariane

[43] IEEE CEDA DATC Robust Design Flow.
https://github.com/ieee-ceda-datc/Robust-Design-Flow

[44] IEEE CEDA DATC Robust Design Flow Calibration.
https://github.com/ieee-ceda-datc/datc-rdf-calibrations

[45] IEEE CEDA Design Automation Technical Committee.
https://ieee-ceda.org/node/2591

[46] IEEE International Workshop on LLM-Aided Design (LAD’24).
https://islad.org

[47] MemPool Group. https://github.com/TILOS-AI-Institute/MacroPlacement/tree/main/Testcases/mempool

[48] MLCAD2024 Artifact Evaluation.
https://github.com/ml-eda/artifact-evaluation

[49] NSF Workshop on Shared Infrastructure for Machine Learning EDA, March 2023.
https://sites.google.com/view/ml4eda/home

[50] OpenCores. https://www.opencores.org/

[51] OpenROAD-Flow-Scripts.
https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts

[52] OpenROAD-Assistant.
https://github.com/OpenROAD-Assistant/OpenROAD-Assistant

[53] OpenRCX. https://github.com/The-OpenROAD-Project/OpenRCX

[54] OpenSTA. https://github.com/The-OpenROAD-Project/OpenSTA

[55] SKY130-PDK. https://github.com/google/skywater-pdk

[56] The OpenROAD Project (GitHub). https://theopenroadproject.org and https://github.com/The-OpenROAD-Project/OpenROAD

[57] TILOS-AI-Institute/MacroPlacement.
https://github.com/TILOS-AI-Institute/MacroPlacement

[58] NVlabs/CircuitOps.
https://github.com/NVlabs/CircuitOps