

# The OpenROAD Project: Unleashing Hardware Innovation

Andrew B. Kahng<sup>\*,†</sup> and Tom Spyrou<sup>\*,‡</sup>

(\*) CSE and (†) ECE Depts., UC San Diego

(‡) Precision Innovations, Inc.

abk@eng.ucsd.edu

aspyrou@eng.ucsd.edu

<https://theopenroadproject.org/>

**Abstract**—The OpenROAD project develops an open-source RTL-to-GDS tool that generates manufacturable layout from a given hardware description – in 24 hours, with no human in the loop. The project is part of the IDEA program within the DARPA ERI. By reducing today’s cost, expertise and schedule barriers to hardware design, OpenROAD enables access to ASIC implementation, thus unleashing hardware innovation. This paper describes the status and outlook for OpenROAD as of its v1.0 release. The OpenROAD tool is integrated around an open-source, commercial-quality database and timing engine. A SkyWater 130nm tapeout was made by ebless.com in May 2020. DRC-clean layout generation in GLOBALFOUNDRIES 12nm was achieved in July 2020. OpenROAD’s futures include (i) serving as a foundation for academic research and teaching; (ii) seeding the transition of open-source EDA into government and commercial usage; and (iii) driving new machine learning research that further accelerates EDA and hardware innovation. With permissively open-sourced code, and no restrictions on sharing of scripts, OpenROAD enables transparency and reproducibility of hardware and EDA research, thus accelerating the pace of discovery.

## I. INTRODUCTION

The OpenROAD project tackles a crisis which has been decades in the making: *Hardware design, and system innovation in hardware, are simply too difficult.* Commercial electronic design automation (EDA) tools have become extremely complex as they evolve to meet the demands of design organizations who create products in leading-edge technology nodes and whose goal is to hyper optimize their designs with significant manual effort. Today, EDA tool and design process outcomes are difficult to predict, and expert tool users are needed. This raises barriers of cost, expertise, and risk to hardware innovation.

One path forward is to automation, with “self-driving” design tools and flows. The OpenROAD project <https://theopenroadproject.org> is part of the DARPA IDEA program, which was launched in June 2018 within the U.S. DARPA Electronics Resurgence Initiative. The IDEA program broadly aims for Hardware Compilers 2.0 – automated generation of manufacturable layout in 24 hours, with no human in the loop, and eventually with no loss of quality of results in Power, Performance or Area. IDEA shifts the focus from tools that squeeze out every last picosecond or microwatt from the

manufacturing technology, to “self-driving” tools that require neither expertise nor complex manually derived tool settings to tape out a working chip.

OpenROAD’s scope is digital IC design: the tool takes Verilog hardware description in, and delivers a merged tapeout-ready GDSII layout database. As described in [1], achieving 24-hour automation requires our project to advance three foundational base technologies: **extreme partitioning** to decompose the design problem into bite-sized chunks; intelligent orchestration of **distributed and parallel optimization** using cloud resources; and **machine learning** to model and predict what will happen when a given tool is run on a given design input with a given target. **Freedoms from choice** are also expected in a no-humans tool, just as a self-driving car will eventually have no steering wheel.

## II. OPENROAD TODAY

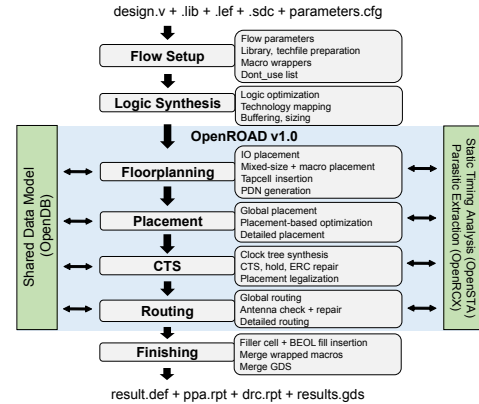


Fig. 1. OpenROAD flow, built around the integrated OpenROAD tool.

Figure 1 depicts OpenROAD’s flow, which is wrapped around our v1.0 integrated tool (aka the “OpenROAD app”). The tool has a modern integrated architecture, industry-strength database and timing analysis, and Tcl and Python scripting interfaces for users. In Spring 2020, a commercial user, Ebless, used OpenROAD for tapeouts of the “striVe” family of SOC designs in the SkyWater 130nm [7] technology. In Summer 2020, OpenROAD achieved a set of “proof points”, including a 12nm SOC tape-in, for automated generation of manufacturable layout (i.e., passing all physical verification checks, and electrical and timing correctness checks) in TSMC 65LP and GLOBALFOUNDRIES 12LP technologies. Figure

DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited. Award number: DARPA HR0011-18-2-0032. The views, opinions and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

2 shows a manufacturable layout produced by OpenROAD for a single-core version of the University of Washington Black-Parrot SOC [15], implemented in GLOBALFOUNDRIES 12LP using Arm 9-track cells and SRAMs, and Invecas IOs.

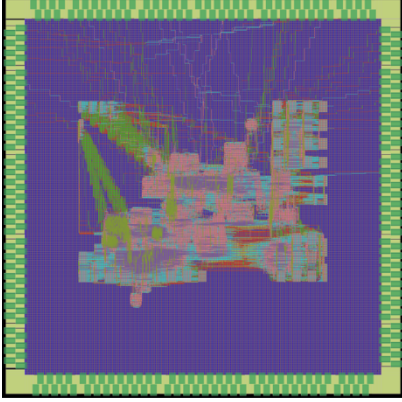


Fig. 2. OpenROAD automated layout of a single-core version of the BlackParrot SOC, in GF12LP technology.

### III. DEVELOPMENT APPROACH: BUILT TO LAST

The OpenROAD team is working to build a platform for research and innovation which can stand the test of time. This includes having a strong software development methodology and testing platform. The team is comprised of student researchers and EDA industry veterans, which allows our effort to distinguish and balance between prototyping for innovation and robust feature development. OpenROAD has a continuous integration framework using Jenkins and GitHub hooks to ensure that no code which breaks the tests can be integrated. Figure 3 shows an example of a report from our public CI.

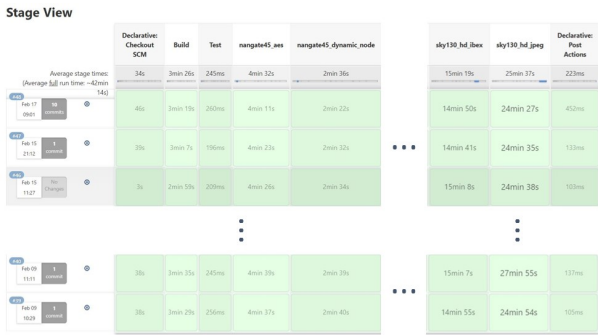


Fig. 3. Public CI snapshot.

Some organizations will entrust the OpenROAD team with confidential data. The team is set up to keep this data secure and private. This secure data is also used to test the system and ensure that the software stays stable for our critical partners with confidential data. Figure 4 shows a snapshot of our secure CI. We do not provide external contributors access to this secure CI since the results are confidential and must remain secure as well. However, infrequent failures found from these contributions are caught by periodic secure testing and are



Fig. 4. Secure CI snapshot.

resolved by internal developers who have both the needed NDA access and the required security training.

OpenROAD's continuous integration-based software development environment allows for parallel development using many branches and repository forks. Our goal is to develop a community of contributors: this has started with members of the project, and is expanding to others who are interested in developing the core technology, as well as to users who are interested in completing designs and adding new methodologies to the flow. Figure 5 shows the security protocols and mechanisms that allow OpenROAD to manage development and data – both secure and public – in its growing community.

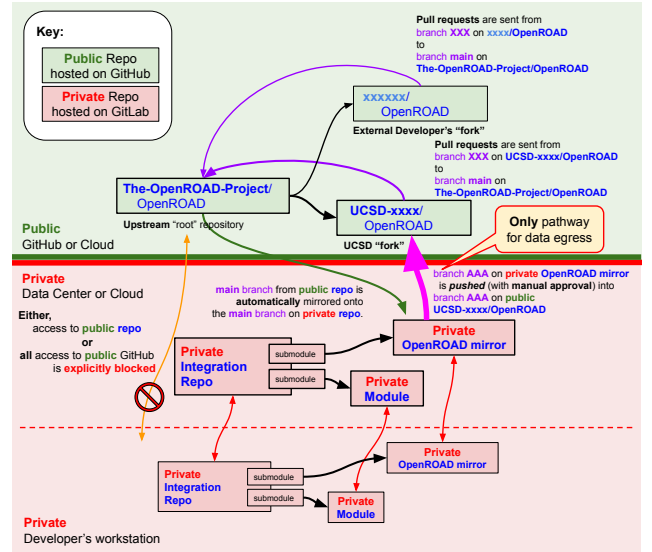


Fig. 5. Secure community interaction flow.

### IV. DESIGN SUCCESSES AND COMMUNITY ENGAGEMENT

As a project, OpenROAD is strongly invested in supporting both industry and academic users, across a broad range of commercial and research design needs. Our overarching goal is to support and engage the community to grow an ecosystem. To this end, the project has organized and sponsored prizes for academic contests (e.g., ICCAD-2019, TAU-2020), organized new workshops and meetings (e.g., the Workshop on Open-Source EDA Technology (WOSET)), and presented tutorials

at numerous meetings (e.g., VLSI Design-2020, DAC-2020). A special session at ICCAD-2020, “Taking Open-Source EDA to the Next Level: From Research to Production IC Design”, focused on OpenROAD and its partnerships [16] [3] [7] [9]. OpenROAD also engages with professional societies and consortia on many fronts. For example, OpenROAD is a member of the CHIPS Alliance [23], and participates in workshops of that organization.

**Efabless OpenLANE and Community Growth.** In the spirit of open source, Efabless has extended the OpenROAD application and flow in its design platform called OpenLANE [7]. OpenLANE is focused on the SKY130 130nm open-source process and PDK from SkyWater Technology Foundry [22]. Efabless has built a community around OpenLANE while working closely with the OpenROAD team. This work has resulted in good extensions to OpenROAD as well as a number of design tapeouts. Figure 6 shows the projects in the recent Google-Efabless SKY130 shuttle (multi-project wafer).

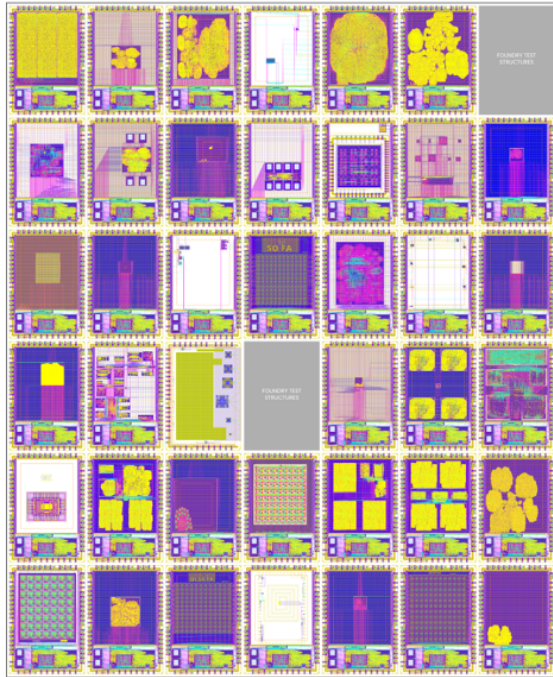


Fig. 6. Recent Google-Efabless SKY130 shuttle with more than 40 designs completed with OpenROAD tooling and Efabless OpenLANE extensions.

The success of OpenLANE and the Fall 2020 release of OpenROAD’s integrated v1.0 tool (see Section V below) are well-correlated with growth of OpenROAD’s user community. Figure 7 shows recent visitors and downloads for OpenROAD, which can be seen as indicators of community engagement.

**ASAP7: An Open-Source, Advanced-Node Research PDK.** To complement the SKY130 open-source manufacturable PDK, OpenROAD has also open-sourced the ASAP7 advanced-node research PDK from Arizona State University [6]. This is a highly realistic PDK with design rules that reflect advanced patterning technologies and scaling boosters (single diffusion break, contact-over-active-gate, dense crossovers, etc.). Cell libraries are released in multiple threshold voltage

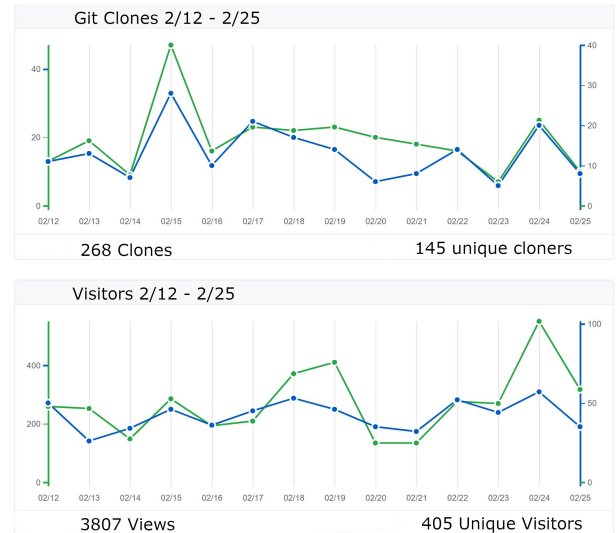


Fig. 7. Community engagement as seen in visitors and downloads.

flavors and with full design enablement. The development roadmap includes memories (SRAM, register files, ROMs, CAMs and TCAMs) and an ASAP5 5nm PDK based on horizontal nanowire transistors. Figure 8 shows example D-latch layouts in the ASAP7 7.5-track and 6-track cell libraries.

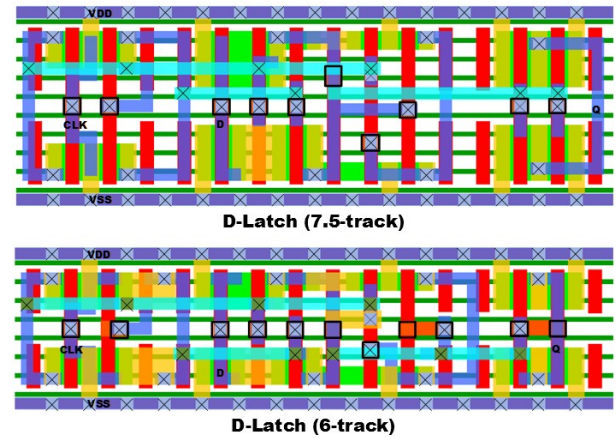


Fig. 8. D-latch layouts in ASAP7. Above: 7.5-track layout. Below: 6-track layout.

**OpenROAD in the IEEE CEDA DATC “Robust Design Flow” (RDF).** Since 2016, the IEEE Council on Electronic Design Automation (CEDA) has, via its Design Automation Technical Committee (DATC), advanced the *Robust Design Flow* (RDF) initiative. Among other objectives, the RDF aims to “facilitate research on flow-scale methodology”; “provide an academic reference flow from logic synthesis to detailed routing”; and “connect academic research to industry practitioners and designs” [4]. OpenROAD has been a key element of the RDF initiative since 2019. The RDF-2020 release includes the OpenROAD integrated app, as well as new *analysis calibrations* that drive progress on timing, power, reliability and other analyses based on OpenROAD results [5].



## V. RECENT ADVANCES

OpenROAD v1.0 is the combined result of many technical and software engineering advances. On the algorithmic end, advances range from more robust sink clustering in clock tree synthesis, to advanced-node design rule handling in the detailed router, to the addition of antenna rule checking and fixing. A *parasitic extraction* tool has been added, and SOC-level planning for wire-bonded and flip-chip packaging is also new in OpenROAD today. Software methodology advances include the robust (Jenkins-based) continuous integration noted earlier, and improved code coverage in tests. Especially noteworthy is the completion of *tight tool integration* onto a single shared data structure. Last, updated logging and metrics infrastructure have enabled machine learning and quality-of-results improvements (power, performance and area – or PPA) to gain traction. This section briefly reviews several highlights.

**Tool Integration.** The integrated v1.0 OpenROAD tool realizes the *incremental shared netlist* architecture that is common to all commercial place-and-route tools since the early 2000s. In Figure 9, the Shared Netlist or Abstract Network Adapter enables incremental netlist modification; it communicates with logic synthesis, place-and-route, clock tree synthesis, post-place optimization, and the static timing engine. The arrows in the figure show how, e.g., if synthesis changes the netlist, there is a callback to place-and-route; if place-and-route updates a cell's location or the routing, this is updated via the Shared Physical or Data Model Adapter, which handles physical information. The timer also listens for a netlist change, and to update delays and slews, the delay calculation must use the updated location and routing information. This architecture enables thousands of sizing or incremental moves per second, and is the heart of optimization in a modern RTL-to-GDS tool.

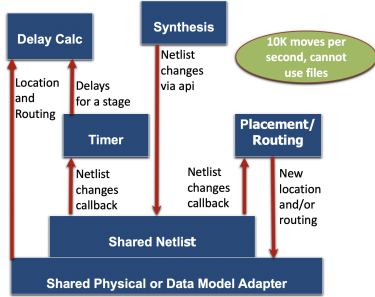


Fig. 9. Incremental shared netlist structure in OpenDB, showing support of incremental optimization via callbacks.

**Parasitic Extraction.** OpenROAD's OpenRCX is a highly scalable, 2.5D extractor optimized for the digital IC design context. It is a product of Athena Design Systems, a mid-2000s era EDA startup whose code base was made available for open-sourcing in late 2019. OpenRCX has been integrated into OpenROAD and validated on 14nm, 28nm, 65nm and 130nm foundry technologies. Figure 10 shows plots from a *jpeg\_encoder* block implemented using OpenROAD in the GF12LP process with an Arm 9-track cell library. The upper part of the figure shows correlations of capacitance and resistance values from OpenRCX versus those reported

by a commercial extraction tool. The bottom part of the figure shows endpoint timing slack correlations between OpenROAD's analysis flow (OpenRCX plus OpenSTA), versus an analogous commercial flow. Note that OpenROAD's analysis stays on the pessimistic side of the 45-degree line, as desired.

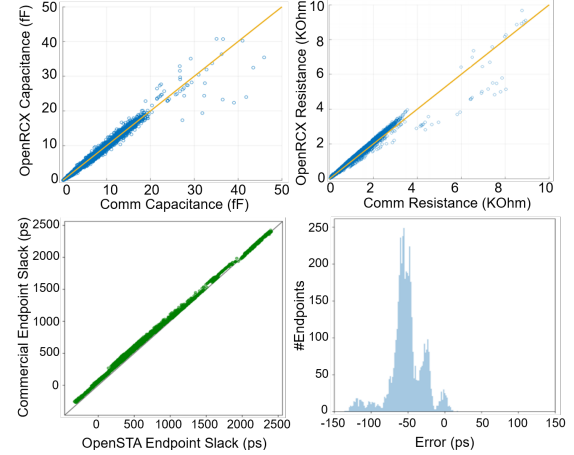


Fig. 10. Correlation of OpenRCX and OpenSTA with commercial parasitic extraction and timing analysis: *jpeg\_encoder* in GF12LP technology.

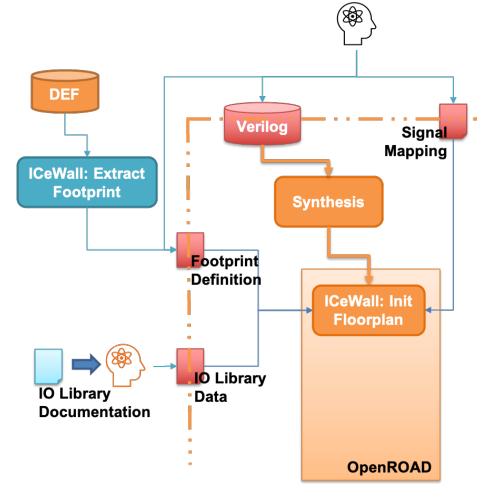


Fig. 11. SOC "footprint" definition and signal mapping in ICeWall.

**Pad Ring and Footprint Creation.** OpenROAD has recently added a capability for pad ring and "footprint" creation in an SOC planning tool, *ICeWall*. A footprint is a pad ring and I/O definition that is reusable across many designs, but that is technology-specific. To facilitate complete automation with less detailed technology-specific knowledge, in each technology a set of footprints will be made available for usage in an automated flow. As shown in Figure 11, ICeWall can take a previous design's layout .def, and extract the footprint from it – or, ICeWall can be used to specify a new footprint from scratch. Users who want full automation will be restricted to the set of predefined footprints. Creation of a new footprint necessarily requires user input and interaction. Furthermore, the I/O definition is often passed down from the board level.

ICeWall has a signal mapping file which can accept this ordering and create a placed pad ring. Figure 11 illustrates the ICeWall workflow.

**GUI.** The development of OpenROAD seeks to achieve critical mass, critical quality, maximum velocity and maximum openness – while attracting a developer community and serving the needs of early-adopter users. This explains why our project, even though it targets 24-hour, no-human-in-the-loop automation, has a GUI. The OpenROAD GUI serves the dual purpose of allowing users to investigate their designs, and developers of the OpenROAD application to investigate their algorithms. Figure 12 shows several images from the OpenROAD GUI.

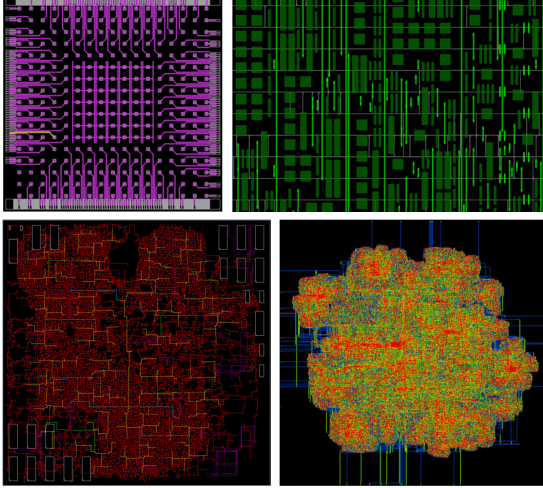


Fig. 12. Images from OpenROAD’s GUI for designs in GF12LP technology. Clockwise from top-left: (i) redistribution layer routing showing 45-degree geometries; (ii) OpenROAD-generated BEOL fill showing OPC and non-OPC fill shapes; (iii) post-global route congestion map for jpeg testcase; (iv) clock tree of single-core BlackParrot testcase.

**Metrics Collection and Machine Learning Enablement.** Machine learning is a key part of OpenROAD’s envisioned no-humans, self-driving RTL-to-GDS capability [10]. To support data collection and data-enabled machine learning, standards and infrastructure are needed [8] [13]. Metrics of the design and the design process must have standard nomenclature and semantics, with a well-defined underlying data model. To this end, OpenROAD provides a working open-source database (OpenDB), data model and implementation. All tools in OpenROAD use a common spdlog-based messaging package, with consistent message types and tool namespaces.<sup>1</sup>

Figure 13 (left) illustrates metrics naming in the current OpenROAD, based on universes of *flow stage* names, *nouns* at run-level and tool metric-level, and *modifiers*. The figure shows how run metrics or tool metrics would have canonical

<sup>1</sup>Standardized metrics collection in OpenROAD flow encompasses both Design metrics (e.g., #buffers, total wirelength) and Run metrics (e.g., CPU time, peak memory usage). This enables not only learning-enabled automation, but continuous improvement of the tool’s quality of results (QoR) as well. Among the purposes that are served by metrics collection: (i) dashboards, and digests of nightly regression runs; (ii) QoR evaluation of incremental functional changes; (iii) validations before any code is merged to the master branch in GitHub; and (iv) large-scale data collection and analysis.

names. For example, the blue path shows the name for post-global placement estimated wirelength. (The metrics can be tied to corners, runs, designs, etc.) In Figure 13 (right), logging in the tool is shown above, and metrics extracted and recorded using Python and JSON are shown below. Using such infrastructure, the research community can collaboratively amass data for machine learning, draw interest via Kaggle contests, etc.

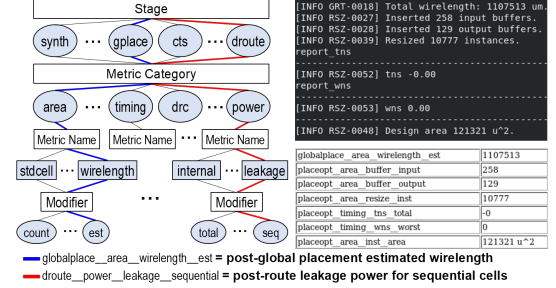


Fig. 13. OpenROAD metrics naming and extraction from logging.

Availability of metrics infrastructure enables analytics and visualizations that help push the tool’s quality of results forward. Using Google Cloud or other compute infrastructure, thousands of results can be gathered in a matter of hours. Figure 14 shows data from a study of the ibex 32-bit RISC-V core in SKY130HS technology. The figure shows that the default runs script in the SKY130HS platform stacks up well against many possible settings, i.e., the yellow circle has good slack and area properties. The colorings show how post-CTS slack (y-axis) is an early indicator of router failure. The use of high-volume experiments to gain insight allows us to ratchet up the baseline results for our testcases.<sup>2</sup>

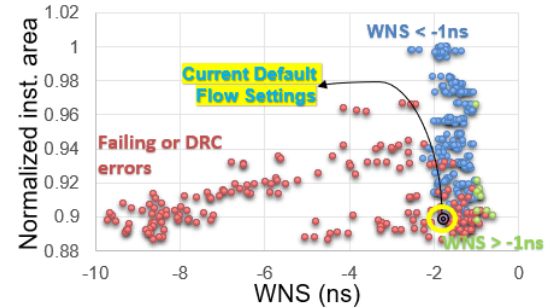


Fig. 14. Visualization of constraint and flow settings for the ibex core in SKY130HS, confirming quality of results from the default OpenROAD runs script.

## VI. LOOKING FORWARD

Over its two-year existence, the OpenROAD project has sought to meet many objectives [12]. (1) It is a four-year basic research project, part of the U.S. DARPA Electronics Resurgence Initiative. (2) It is an RTL-to-GDS EDA tool with external users who use it for new design tapeouts. (3) It is

<sup>2</sup>The single-core BlackParrot layout shown in Figure 2 represents a 36% faster implementation, with 12% less wirelength, compared to what the tool could deliver a few months earlier.

a seed for open-source EDA innovation and research [5] that must reach critical mass and critical quality. And (4) it aligns with interests of a large and active open-source hardware community. Of necessity, OpenROAD has remained agile and flexible, e.g., balancing “production quality” and advanced-node capability against the academic research setting, or balancing no-humans automation against users’ requests for more controllability.

**Focus on Users in Government and the DIB.** Users of EDA technology in the U.S. Government and the defense industrial base (DIB) have in various ways been underserved by traditional EDA companies. The EDA business model (licensing, pricing, support, etc.) and structure (three dominant vendors) places a focus on large contracts for a small number of worldwide enterprise customers. Smaller design organizations, and teams with niche design requirements (reliability, security, heterogeneous integration, novel device and circuit fabrics, as well as extreme ease of tool bringup) are perfect users of OpenROAD. The code is open-source and can be modified as needed. There are no up-front licensing costs. A number of community members can be engaged to provide support, modifications or enhancements to the tool. There is a strong analogy with Linux and its open-source nature which allows the usage of OpenROAD to have flexibility in tandem with robust support for real-world business applications.

As described above, there is currently an open-source manufacturable PDK for the SkyWater 130nm process node. Dozens of tapeouts have been made on this platform using OpenROAD. As the open-source EDA effort evolves, it is only natural that foundries will open-source their PDKs in order to lower the barrier for experimentation by potential customers. OpenROAD supports multiple closed-source commercial PDKs as well. It is exciting to see this development taking place and we expect increasing numbers of PDKs to become open source in the future. The ability to download, design, and manufacture without signing IP agreements or explain design objectives to vendors is an important benefit, especially when secrecy is needed.

**Key Directions.** Looking ahead, two directions are particularly critical to OpenROAD. First, we feel that differentiating, high-value use cases for open-source EDA must be identified. For example, OpenROAD’s low adoption cost and cloud scalability may match well with the long-standing challenge of early system/architecture design space exploration [12]. Second, a sustainable open-source EDA based ecosystem must be created that serves not only IC and system companies, but smaller design teams (e.g., in government and in the DIB) whose needs are not presently well-served, along with open-source hardware and software communities, as well. The interests of EDA researchers and professional societies, policy-making bodies and consortia, and commercial EDA vendors must also be considered. In his recent 2020 DARPA ERI Summit plenary talk [14], the IDEA program manager, Mr. Serge Leef, posed the question: “Can DARPA improve access [to state-of-the-art EDA tools] and fuel advances through open-source EDA technologies?” Finding answers to this question will be important to OpenROAD even as we now pursue “Phase 2” of our project.

## VII. ACKNOWLEDGMENTS

The authors thank the members of the OpenROAD team for tremendous effort and dedication. Thanks are also due to Tim Ansell of Google, Mohamed Kassem of Efabless, and many others in the community of OpenROAD users and contributors. We also owe a great debt to the DARPA IDEA program for supporting the OpenROAD vision.

## REFERENCES

- [1] T. Ajayi, D. Blaauw, T.-B. Chan, C.-K. Cheng, V. A. Chhabria, D. K. Choo, M. Coltella, R. Dreslinski, Mateus Fogaça, S. Hashemi, A. Ibrahim, A. B. Kahng, M. Kim, J. Li, Z. Liang, U. Mallappa, P. Penzes, G. Pradipta, S. Reda, A. Rovinski, K. Samadi, S. S. Sapatnekar, L. Saul, C. Sechen, V. Srinivas, W. Swartz, D. Sylvester, D. Urquhart, L. Wang, M. Woo and B. Xu, “OpenROAD: Toward a Self-Driving, Open-Source Digital Layout Implementation Tool Chain”, *Proc. GOMACTech*, 2019, pp. 1105-1110.
- [2] T. Ajayi, V. A. Chhabria, Mateus Fogaça, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem, G. Pradipta, S. Reda, M. Saligane, S. S. Sapatnekar, C. Sechen, M. Shalan, W. Swartz, L. Wang, Z. Wang, M. Woo and B. Xu, “Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project”, *Proc. DAC*, 2019, pp. 1-4.
- [3] T. Ansell and M. Saligane, “The Missing Pieces of Open Design Enablement: A Recent History of Google Efforts”, *Proc. ICCAD* 2020, Article No. 112, pp. 1-8.
- [4] 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.
- [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-2020: Strengthening the Foundation for Academic Research in IC Physical Design”, *Proc. ICCAD*, 2020. <https://github.com/iee-cceda-datc/datc-robust-design-flow>
- [6] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy and G. Yeric, “ASAP7: A 7-nm FinFET Predictive Process Design Kit”, *Microelectronics J.* 53 (2016), pp. 105–115.
- [7] T. Edwards and M. Shalan, “Building OpenLane: A 130nm OpenROAD-based Tapeout-Proven Flow”, *Proc. ICCAD*, 2020, Article No. 110, pp. 1-6.
- [8] S. Fenstermaker, D. George, A. B. Kahng, S. Mantik and B. Thielges, “METRICS: A System Architecture for Design Process Optimization”, *Proc. DAC*, 2000, pp. 705-710.
- [9] M. Fogaça, E. Monteiro, M. Danigno, I. Oliveira, P. Butzen and R. Reis, “Contributions to OpenROAD from Abroad: Experiences and Learnings”, *Proc. ICCAD* 2020, Article No. 113, pp. 1-8.
- [10] A. B. Kahng, “Machine Learning Applications in Physical Design: Recent Results and Directions”, *Proc. ISPD*, 2018, pp. 68-73.
- [11] A. B. Kahng, “Looking Into the Mirror of Open Source”, *Proc. ICCAD*, 2019.
- [12] A. B. Kahng, “Open-Source EDA: If We Build It, Who Will Come?”, *Proc. 28th IFIP/IEEE Intl. Conf. on Very Large Scale Integration*, 2020.
- [13] A. B. Kahng and S. Mantik, “A System for Automatic Recording and Prediction of Design Quality Metrics”, *Proc. ISQED*, 2001, pp. 81-86.
- [14] S. Leef, “Open Source Accelerated Chip Design”, plenary talk, *DARPA ERI Summit*, August 18, 2020.
- [15] D. Petrisko, F. Gilani, M. Wyse, D. C. Jung, S. Davidson, P. Gao, C. Zhao, Z. Azad, S. Canakci, B. Veluri, T. Guirino, A. Joshi, M. Oskin and M. B. Taylor, “BlackParrot: An Agile Open-Source RISC-V Multicore for Accelerator SoCs”, *IEEE Micro* 40(4) (2020), pp. 93-102.
- [16] A. Rovinski, T. Ajayi, M. Kim, G. Wang, and M. Saligane, “Bridging Academic Open-Source EDA to Real-World Usability”, *Proc. ICCAD* 2020, Article No. 111, pp. 1-7.
- [17] DATC-RDF, <https://github.com/iee-cceda-datc/datc-robust-design-flow>
- [18] DARPA IDEA Program, <https://www.darpa.mil/program/intelligent-design-of-electronic-assets>
- [19] OpenROAD website, <https://theopenroadproject.org/>
- [20] OpenROAD Tool GitHub, <https://github.com/The-OpenROAD-Project/OpenROAD>
- [21] ASAP7 PDK and 7.5-Track Cell Library, <https://github.com/The-OpenROAD-Project/asap7>
- [22] SkyWater 130nm Open Source PDK, <https://github.com/google/skywater-pdk>
- [23] CHIPS (Common Hardware for Interfaces, Processors and Systems) Alliance, <https://chipsalliance.org/>