# How I Learned to Stop Worrying and Love Physical Design

Edward Wang
Massachusetts Institute of Technology
USA

Luca Daniel
Massachusetts Institute of Technology
USA

Yoni Zohar
Bar-Ilan University
Israel

Clark Barrett
Stanford University
USA

## ABSTRACT

Silicon prototypes (tapeouts) are crucial in realizing new hardware accelerator designs, yet physical design continues to be a formidable bottleneck which prevents more designs from reaching tapeout. We explore some possible factors behind the difficulty and speculate on some paths forward.

## 1 INTRODUCTION

Silicon prototypes (or tapeouts) play a crucial role in the development and evaluation of new hardware accelerator architectures. These prototypes validate architectural assumptions and decisions with the highest fidelity performance and power results [36]. This is particularly important as some architectural limitations might only surface during physical design when issues such as congestion, routability, and design violations arise [29].

Agile hardware design is a highly promising approach for addressing the challenges of hardware design. It is an adaptation of agile software development centred around regular cycles which run through all the phases of hardware design, culminating in a tape-in[1] that reflects all stages of the hardware design process [24]. This approach takes advantage of tool-focused advances such as [6][4] to enable small teams to more efficiently deliver a design ready for tapeout. [24] is a successfully applied example of agile hardware design which has produced numerous experimental RISC-V silicon prototypes [4].

However, physical design remains a major bottleneck in agile hardware design. Having one step take much much longer than other steps in the process throws agile off-balance - agile tape-ins become increasingly bottlenecked on the physical design before collapsing to a waterfall-like phase in the last few weeks before tapeout as all efforts become focused on physical design [37][27]. This issue greatly impedes the progress of hardware accelerator development as many new proposals do not make it to silicon due to the long and intensive process of physical design. As a result, the critical bottleneck of physical design continues to hinder the advancement of hardware design and prevents the production of high-fidelity results to guide future architectural research.

## 2 WHY IS PHYSICAL DESIGN SO CHALLENGING?

### 2.1 Factor 1: classical physical design flows are deficient

Classical physical design flows[2] (see Figure 1) suggest that designing a usable physical layout is a merely matter of running some tools once with straightforward parameters [37]. This simplistic view, however, fails to take into account the many complexities involved in this process. A slightly more representative flow is shown in Figure 2. Furthermore, typical flows intermix distinct information about the design, the physical implementation, the CAD tool, and the PDK,[3] making physical design re-use difficult. Additionally, typical flows do not empower the user to iteratively run tools multiple times in order to get a correct layout [37]. These factors result in a painful experience of even starting to use physical design tools in a project.

### 2.2 Factor 2: place-and-route is algorithmically hard

Place-and-route is considered to be an NP-hard problem [21] [35]. While NP-complete problems can be found in software compilation, often the NP-completeness in those problems can be avoided in practice. For example, while register allocation is frequently posed as an NP-complete problem, its classical analysis yields NP-completeness if and only if no spilling [4] are allowed [10]. In practice, many compilers do not aim for spill-free compilations [30], greatly reducing the computational complexity of the problem while still providing a functional compilation output.

However, this difficulty cannot be easily avoided in place-and-route, as violations will result in unmanufacturable or non-functional circuits. A mismatch of expectation materialises when place-and-route tools are advertised as "compilers"[5]; we expect them to provide usable (if not sub-optimal) outputs that do not require manual intervention [37][27]. The analogy in software would be needing to manually fix assembly generated after running gcc/clang. In short, there is unlikely to be a "silver bullet" in physical design. [37].

[1] A tape-in is an intermediate-stage fabricatable design that may be missing features. It is the hardware analogue of an agile software release, designed to encourage responsive, continuous development style over large monolithic leaps [24].

[2] Also referred to as "VLSI flows".

[3] Process design kit: a set of technical information provided by a semiconductor foundry to enable designers to target a particular manufacturing process.

[4] And no control flow graph modifications either.

[5] e.g. IC Compiler

*LATTE '24, April 27, 2024, San Diego, CA, USA*

**Figure 1: An "advertised" VLSI flow [37][27]**

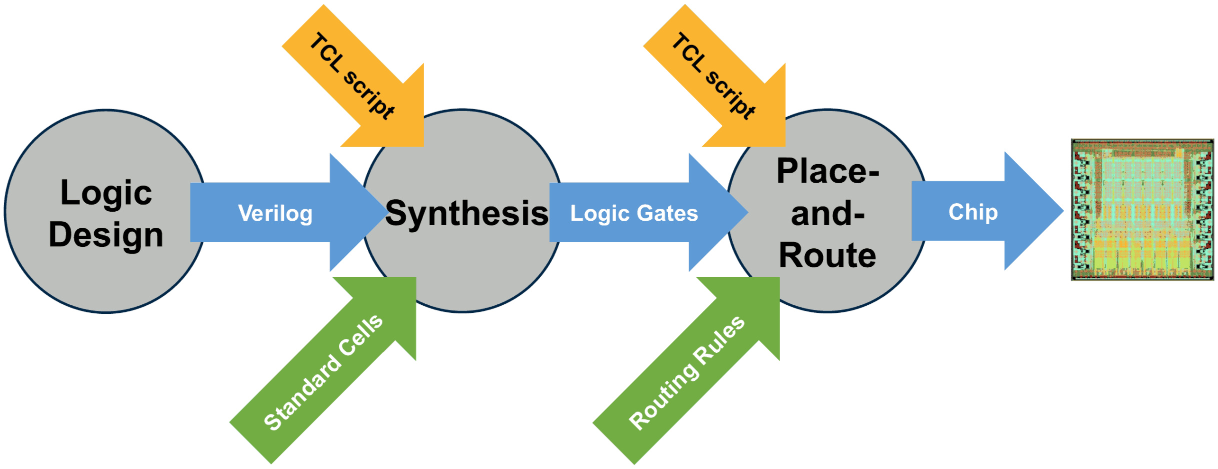
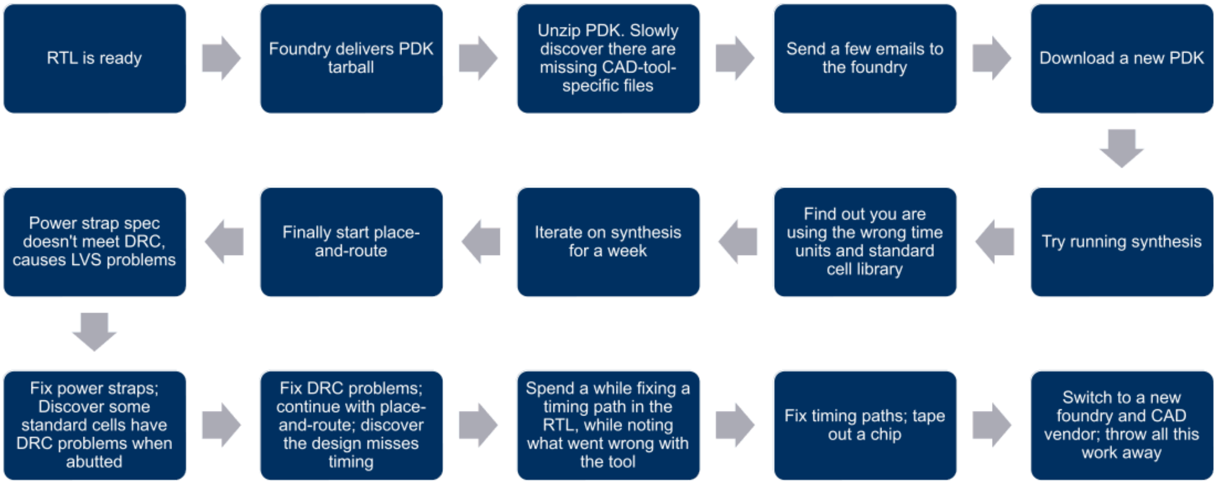


**Figure 2: A more realistic (yet still simplified) VLSI flow [37][27]**


RTL is ready
Foundry delivers PDK tarball
Unzip PDK. Slowly discover there are missing CAD-tool-specific files
Send a few emails to the foundry
Download a new PDK
Try running synthesis
Find out you are using the wrong time units and standard cell library
Iterate on synthesis for a week
Finally start place-and-route
Power strap spec doesn't meet DRC, causes LVS problems
Fix power straps; Discover some standard cells have DRC problems when abutted
Fix DRC problems; continue with place-and-route; discover the design misses timing
Spend a while fixing a timing path in the RTL, while noting what went wrong with the tool
Fix timing paths; tape out a chip
Switch to a new foundry and CAD vendor; throw all this work away

### 2.3 Factor 3: dominant approaches are not correct-by-construction

Many current place-and-route algorithmic approaches are primarily heuristic/statistical-based in order to address performance challenges [18]. [6] These approaches work by converging on increasingly correct results through iterative means or randomness. However, due to the non-convex nature of NP-hard problems [17], these methods cannot guarantee a correct layout, leaving room for potential errors/violations.

Finally, even specialized place-and-route algorithms often make sacrifices to address performance challenges, leaving violations in the final layout [34][12]. This situation is both time-consuming and frustrating for hardware designers, as they are often forced to spend a significant amount of time and effort fixing these violations manually [37][27][8][31].

### 2.4 Factor 4: software engineering methodologies for CAD are limited

Despite the importance of CAD such as place-and-route in hardware design, the current state of CAD software engineering presents several challenges and limitations. A major challenge is that place-and-route CAD tools are typically developed using performance-engineered C/C++ (e.g. [19]). While these languages are chosen for their ability to deliver high performance, it has been shown that they can be difficult to maintain, optimize, and verify [32][25]. [7] This can lead to an increase in security vulnerabilities in often-complex CAD software, as exemplified by [5]. This highlights a need for exploring alternative approaches to address these challenges and mitigate the potential for security vulnerabilities.

[6] Statistical approaches include simulated annealing and spreading; iterative approaches include legalization, rip-up-and-replace [18].

[7] Additionally, the closed-source nature of some popular CAD tools and the economics of chip design may also play a role.

## 3 THE WAY FORWARD?

We present a few viewpoints from which to view the current physical design tooling situation.

The first viewpoint maintains that the current state of place-and-route in physical design is sufficient. It also means that realizing silicon prototypes will by and large remain inaccessible to most (small teams of) designers. Physical designers would continue to be hindered by needing to manually fix design violations generated by tools.[8] Additionally, the challenges associated with current tooling create a disincentive for new engineers and researchers to enter hardware design, leading to an uncertain future for the field as a whole [15]. Finally, this creates a disincentive as upstream advancements can become bottlenecked in physical design, leaving the road to agile hardware design incomplete.

The second viewpoint suggests to invest into better system-level tools. Tools such as Hammer [37][27] address many systems-level challenges. Other approaches include FuseSoC/Edalize [20] and Mflowgen [11] among others. Chipyard integrates the above work into an architectural/RTL-level generator [3]. While these projects help lower the barrier for physical design, they do not address deficiencies and issues with the underlying tools. Additionally, they are unable to provide significant insight into the internal workings of CAD tools, limiting opportunities for further research.

The third viewpoint is to invest into underlying open source tools. For example, OpenROAD [1] has made signficant inroads towards a complete and usable open source flow similar to ICC or Innovus. This would be an essential piece of infrastructure for open source tools that would also greatly lower the barrier for entering into physical design, especially if integrated with system-level tools. Challenges include, if viewed from merely the open source angle without incorporating novel methodologies, lack of ecosystem investment, high expectations, security and correctness concerns [1].

A fourth viewpoint looks for alternative methodologies instead of traditional place-and-route techniques. One popular approach is to forgo place-and-route entirely and instead use expert insight to write scripts or tools to generate layouts. This approach has been adopted in projects such as [13][16]. However, these systems are very time-consuming to use and do not guarantee correctness. More automated tools could open opportunities for co-optimization that can be missed by hand-written layout systems.

Another alternative methodology would be to leverage formal methods. Given the explosion in design complexity and rules, it has become increasingly difficult to ensure that correctness rules are met in conjunction with the increased demands on PPA[9] [26]. Despite the impressive recent advances in AI/ML, significant concerns remain around reliability and trustworthiness which pose verification challenges [23]. While formal methods have been known for poor performance compared to traditional algorithm-based approaches, recent advances in SMT solving have been pushing the frontier [7][14]. They have shown promise in other domains including web layout engines [28], dungeon generation [38], and automotive plant layouts [9] which serves as a source of inspiration.

[8] For example: references to all-nighters in [22][2][33]

[9] "Power, performance, area" is a shorthand for evaluation metrics for hardware designs.

## 4 ACKNOWLEDGEMENTS

The authors would like to thank Richard Lin for his review and input on the manuscript.

## REFERENCES

[1] Tutu Ajayi, Vidya A. Chhabria, Mateus Fogaça, Soheil Hashemi, Abdelrahman Hosny, Andrew B. Kahng, Minsoo Kim, Jeongsup Lee, Uday Mallappa, Marina Neseem, Geraldo Pradipta, Sherief Reda, Mehdi Saligane, Sachin S. Sapatnekar, Carl Sechen, Mohamed Shalan, William Swartz, Lutong Wang, Zhehong Wang, Mingyu Woo, and Bangqi Xu. 2019. Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project. In *Proceedings of the 56th Annual Design Automation Conference 2019* (Las Vegas, NV, USA) *(DAC '19)*. Association for Computing Machinery, New York, NY, USA, Article 76, 4 pages. https://doi.org/10.1145/3316781.3326334

[2] Tarek Al Abbas. 2019. Miniature high dynamic range time-resolved CMOS SPAD image sensors. (2019). http://core.ac.uk/download/pdf/429736111.pdf

[3] Alon Amid, David Biancolin, Abraham Gonzalez, Daniel Grubb, Sagar Karandikar, Harrison Liew, Albert Magyar, Howard Mao, Albert Ou, Nathan Pemberton, Paul Rigge, Colin Schmidt, John Wright, Jerry Zhao, Yakun Sophia Shao, Krste Asanović, and Borivoje Nikolić. 2020. Chipyard: Integrated Design, Simulation, and Implementation Framework for Custom SoCs. *IEEE Micro* 40, 4 (2020), 10–21. https://doi.org/10.1109/MM.2020.2996616

[4] Krste Asanovic, Rimas Avizienis, Jonathan Bachrach, Scott Beamer, David Biancolin, Christopher Celio, Henry Cook, Daniel Dabbelt, John Hauser, Adam Izraelevitz, et al. 2016. The rocket chip generator. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17* 4 (2016), 6–2. http://aspire.eecs.berkeley.edu/wp/wp-content/uploads/2016/04/Tech-Report-The-Rocket-Chip-Generator-Beamer.pdf

[5] Autodesk. 2023. Multiple Vulnerabilities in the Autodesk AutoCAD and Maya Desktop Software. http://web.archive.org/web/20231211081923/https://www.autodesk.com/trust/security-advisories/adsk-sa-2022-0020. http://web.archive.org/web/20231211081923/https://www.autodesk.com/trust/security-advisories/adsk-sa-2022-0020

[6] Jonathan Bachrach, Huy Vo, Brian Richards, Yunsup Lee, Andrew Waterman, Rimas Avižienis, John Wawrzynek, and Krste Asanović. 2012. *Chisel: constructing hardware in a Scala embedded language.* ACM, 1216–1225.

[7] Haniel Barbosa, Clark Barrett, Martin Brain, Gereon Kremer, Hanna Lachnitt, Makai Mann, Abdalrhman Mohamed, Mudathir Mohamed, Aina Niemetz, Andres Nötzli, Alex Ozdemir, Mathias Preiner, Andrew Reynolds, Ying Sheng, Cesare Tinelli, and Yoni Zohar. 2022. cvc5: A Versatile and Industrial-Strength SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, Dana Fisman and Grigore Rosu (Eds.). Springer International Publishing, Cham, 415–442.

[8] Mudasir Bashir, Fatemeh Abbassi, Mirjana Videnovic Misic, Johannes Sturm, and Hueber Getnot. 2020. Performance Comparison of BAG and Custom Generated Analog Layout for Single-Tail Dynamic Comparator. In *2020 Austrochip Workshop on Microelectronics (Austrochip)*. 37–41. https://doi.org/10.1109/Austrochip51129.2020.9232979

[9] Nikolaj Bjørner, Maxwell Levatich, Nuno P. Lopes, Andrey Rybalchenko, and Chandrasekar Vuppalapati. 2021. Supercharging Plant Configurations Using Z3. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, Peter J. Stuckey (Ed.). Springer International Publishing, Cham, 1–25.

[10] Florent Bouchez, Alain Darte, Christophe Guillon, and Fabrice Rastello. 2007. Register Allocation: What Does the NP-Completeness Proof of Chaitin et al. Really Prove? Or Revisiting Register Allocation: Why and How. In *Languages and Compilers for Parallel Computing*, George Almási, Călin Caşcaval, and Peng Wu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 283–298.

[11] Alex Carsello, James Thomas, Ankita Nayak, Po-Han Chen, Mark Horowitz, Priyanka Raina, and Christopher Torng. 2022. mflowgen: a modular flow generator and ecosystem for community-driven physical design: invited. In *Proceedings of the 59th ACM/IEEE Design Automation Conference* (San Francisco, California) *(DAC '22)*. Association for Computing Machinery, New York, NY, USA, 1339–1342. https://doi.org/10.1145/3489517.3530633

[12] William Chow, Gracieli Posser, Stefanus Mantik, Yixiao Ding, and Wen-Hao Liu. 2018. ISPD19 Contest: Evaluation Metrics and Ranking Method. http://web.archive.org/web/20221203032521/https://ispd.cc/contests/19/metrics_and_ranking.pdf. http://web.archive.org/web/20221203032521/https://ispd.cc/contests/19/metrics_and_ranking.pdf

[13] J. Crossley, A. Puggelli, H.-P. Le, B. Yang, R. Nancollas, K. Jung, L. Kong, N. Narevsky, Y. Lu, N. Sutardja, E. J. An, A. L. Sangiovanni-Vincentelli, and E. Alon. 2013. BAG: A designer-oriented integrated framework for the development of AMS circuit generators. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 74–81. https://doi.org/10.1109/ICCAD.2013.6691100

[14] Cristina David and Daniel Kroening. 2017. Program synthesis: challenges and opportunities. *Philosophical Transactions of the Royal Society A: Mathematical,*

*Physical and Engineering Sciences* 375, 2104 (2017), 20150403. https://doi.org/10.1098/rsta.2015.0403

[15] Tom Dillinger. 2022. A Crisis in Engineering Education - Where are the Microelectronics Engineers. http://web.archive.org/web/20231208093723/https://semiwiki.com/events/314964-a-crisis-in-engineering-education-where-are-the-microelectronics-engineers/. http://web.archive.org/web/20231208093723/https://semiwiki.com/events/314964-a-crisis-in-engineering-education-where-are-the-microelectronics-engineers/

[16] Jaeduk Han, Woorham Bae, Eric Chang, Zhongkai Wang, Borivoje Nikolić, and Elad Alon. 2021. LAYGO: A Template-and-Grid-Based Layout Generation Engine for Advanced CMOS Technologies. *IEEE Transactions on Circuits and Systems I: Regular Papers* 68, 3 (2021), 1012–1022. https://doi.org/10.1109/TCSI.2020.3046524

[17] Prateek Jain and Purushottam Kar. 2017. Non-convex Optimization for Machine Learning. *Foundations and Trends® in Machine Learning* 10, 3-4 (2017), 142–363. https://doi.org/10.1561/2200000058

[18] Andrew B Kahng, Jens Lienig, Igor L Markov, and Jin Hu. 2011. *VLSI physical design: from graph partitioning to timing closure*. Vol. 312. Springer.

[19] Andrew B. Kahng, Lutong Wang, and Bangqi Xu. 2021. TritonRoute: The Open-Source Detailed Router. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40, 3 (2021), 547–559. https://doi.org/10.1109/TCAD.2020.3003234

[20] Olof Kindgren. 2019. A scalable approach to IP management with FuseSoC. In *1st Workshop on Open-Source Design Automation (OSDA)*. http://fusesoc.net/osda19_fusesoc.pdf

[21] Mark R. Kramer and Jan Van Leeuwen. 1982. *Wire Routing is NP-Complete*. Technical Report RUU-CS-82-4. Department of Computer Science, University of Utrecht. http://web.archive.org/web/20180504191319/https://dspace.library.uu.nl/bitstream/handle/1874/16302/kramer_82_wire_routing.pdf

[22] Rajesh Kumar. 2003. Interconnect and noise immunity design for the Pentium 4 processor. In *Proceedings of the 40th Annual Design Automation Conference* (Anaheim, CA, USA) *(DAC '03)*. Association for Computing Machinery, New York, NY, USA, 938–943. https://doi.org/10.1145/775832.776068

[23] Kim Larsen, Axel Legay, Gerrit Nolte, Maximilian Schlüter, Marielle Stoelinga, and Bernhard Steffen. 2022. Formal Methods Meet Machine Learning (F3ML). In *Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning*, Tiziana Margaria and Bernhard Steffen (Eds.). Springer Nature Switzerland, Cham, 393–405.

[24] Yunsup Lee, Andrew Waterman, Henry Cook, Brian Zimmer, Ben Keller, Alberto Puggelli, Jaehwa Kwak, Ruzica Jevtic, Stevo Bailey, Milovan Blagojevic, Pi-Feng Chiu, Rimas Avizienis, Brian Richards, Jonathan Bachrach, David Patterson, Elad Alon, Bora Nikolic, and Krste Asanovic. 2016. An Agile Approach to Building RISC-V Microprocessors. *IEEE Micro* 36, 2 (March 2016), 8–20. https://doi.org/10.1109/mm.2016.11

[25] Xavier Leroy, Sandrine Blazy, Daniel Kästner, Bernhard Schommer, Markus Pister, and Christian Ferdinand. 2016. CompCert-a formally verified optimizing compiler. In *ERTS 2016: Embedded Real Time Software and Systems, 8th European Congress*.

[26] Helen Li, Robben Lee, Tyzy Lee, Teddy Xue, Hermes Liu, Hall Wu, Qijian Wan, Chunshan Du, Xinyi Hu, and Zhengfang Liu. 2018. An efficient way of layout processing based on calibre DRC and pattern matching for defects inspection application. In *Design-Process-Technology Co-optimization for Manufacturability XII*, Jason P. Cain (Ed.), Vol. 10588. International Society for Optics and Photonics, SPIE, 105880Y. https://doi.org/10.1117/12.2297349

[27] Harrison Liew, Daniel Grubb, John Wright, Colin Schmidt, Nayiri Krzysztofowicz, Adam Izraelevitz, Edward Wang, Krste Asanović, Jonathan Bachrach, and Borivoje Nikolić. 2022. Hammer: a modular and reusable physical design flow tool: invited. In *Proceedings of the 59th ACM/IEEE Design Automation Conference* (San Francisco, California) *(DAC '22)*. Association for Computing Machinery, New York, NY, USA, 1335–1338. https://doi.org/10.1145/3489517.3530672

[28] Junrui Liu, Yanju Chen, Eric Atkinson, Yu Feng, and Rastislav Bodik. 2023. Conflict-Driven Synthesis for Layout Engines. *Proc. ACM Program. Lang.* 7, PLDI, Article 132 (jun 2023), 22 pages. https://doi.org/10.1145/3591246

[29] Yen-Fu Liu, Chou-Ying Hsieh, and Sy-Yen Kuo. 2023. Boomerang: Physical-Aware Design Space Exploration Framework on RISC-V SonicBOOM Microarchitecture. In *2023 IEEE 34th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. 85–93. https://doi.org/10.1109/ASAP57973.2023.00026

[30] Jakob Stoklund Olesen. 2011. Greedy Register Allocation in LLVM 3.0. http://blog.llvm.org/2011/09/greedy-register-allocation-in-llvm-30.html. http://blog.llvm.org/2011/09/greedy-register-allocation-in-llvm-30.html

[31] Qu1ck. 2020. Why do people choose to NOT use the auto-router. http://web.archive.org/web/20240307204236/https://forum.kicad.info/t/why-do-people-choose-to-not-use-the-auto-router/25849/2. http://web.archive.org/web/20240307204236/https://forum.kicad.info/t/why-do-people-choose-to-not-use-the-auto-router/25849/2

[32] Jonathan Ragan-Kelley, Connelly Barnes, Andrew Adams, Sylvain Paris, Frédo Durand, and Saman Amarasinghe. 2013. Halide: a language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Seattle, Washington, USA) *(PLDI '13)*. Association for Computing Machinery, New York, NY, USA, 519–530. https://doi.org/10.1145/2491956.2462176

[33] Ajith Sivadhasan Ramani. 2017. *A differential push-pull voltage mode driver for vertical-cavity surface emitting laser*. Ph.D. Dissertation. University of British Columbia. https://doi.org/10.14288/1.0362596

[34] Fan-Keng Sun, Hao Chen, Ching-Yu Chen, Chen-Hao Hsu, and Yao-Wen Chang. 2018. A Multithreaded Initial Detailed Routing Algorithm Considering Global Routing Guides. In *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 1–7. https://doi.org/10.1145/3240765.3240777

[35] Atsushi Takahashi. 2013. Dawn of computer-aided design: from graph-theory to place and route. In *Proceedings of the 2013 ACM International Symposium on Physical Design* (Stateline, Nevada, USA) *(ISPD '13)*. Association for Computing Machinery, New York, NY, USA, 58. https://doi.org/10.1145/2451916.2451930

[36] Christopher Torng, Shunning Jiang, Khalid Al-Hawaj, Ivan Bukreyev, Berkin Ilbeyi, Tuan Ta, Lin Cheng, Julian Puscar, Ian Galton, and Christopher Batten. 2018. A New Era of Silicon Prototyping in Computer Architecture Research. In *The RISC-V Day Workshop at the 51st Int'l Symp. on Microarchitecture*. http://www.csl.cornell.edu/~cbatten/pdfs/torng-brgtc2-riscvday2018.pdf

[37] Edward Wang, Colin Schmidt, Adam Izraelevitz, John Wright, Borivoje Nikolić, Elad Alon, and Jonathan Bachrach. 2020. A Methodology for Reusable Physical Design. In *2020 21st International Symposium on Quality Electronic Design (ISQED)*. 243–249. https://doi.org/10.1109/ISQED48828.2020.9136999

[38] Jim Whitehead. 2020. Spatial Layout of Procedural Dungeons Using Linear Constraints and SMT Solvers. In *Proceedings of the 15th International Conference on the Foundations of Digital Games* (Bugibba, Malta) *(FDG '20)*. Association for Computing Machinery, New York, NY, USA, Article 101, 9 pages. https://doi.org/10.1145/3402942.3409603