A Case for Open EDA Verticals

Zhiru Zhang
zhiruz@cornell.edu
Cornell University
Ithaca, NY, USA

Matthew Hofmann
mrh259@cornell.edu
Cornell University
Ithaca, NY, USA

Andrew Butt
atb78@cornell.edu
Cornell University
Ithaca, NY, USA

ABSTRACT

With the end of Dennard scaling and Moore's Law reaching its limits, domain-specific hardware specialization has become a crucial method for improving compute performance and efficiency for various important applications. Leading companies in competitive fields, such as machine learning and video processing, are building their own in-house technology stacks to better suit their accelerator design needs. However, currently this approach is only a viable option for a few large enterprises that can afford to invest in teams of experts in hardware, systems, and compiler development for high-value applications. In particular, the high license cost of commercial electronic design automation (EDA) tools presents a significant barrier for small and mid-size engineering teams to create new hardware accelerators. These tools are essential for designing, simulating, and testing new hardware, but can be too expensive for smaller teams with limited budgets, reducing their ability to innovate and compete with larger organizations.

More recently, open-source EDA toolflows [1] [8] [7] [3] have emerged which offer a promising alternative to commercial tools, with the potential to provide more cost-effective solutions for hardware development. For example, OpenROAD [1] allows the design of custom ASICs with minimal human intervention and no licensing fees. During initial development, it was also able to take advantage of existing tools such as Yosys [**?**] and KLayout [4] to reduce the amount of new code required to get a working flow. However, early adoption of open-source alternatives carries risk, as open-source EDA projects often lack important features and are less reliable than commercial options. Additionally, current open-source EDA tools may produce less competitive quality of results (QoR) and may not be able to catch up to commercial solutions anytime soon. Even when EDA tool access is not an issue, designing and implementing special-purpose accelerators using conventional RTL methodology can be unproductive and incurs high non-recurring engineering (NRE) costs. High-level synthesis (HLS) has become increasingly popular in both academia and industry to automatically generate RTL designs from software programs. However, existing HLS tools do not help maintain domain-specific context throughout the design flow (e.g., placement, routing), which makes achieving good QoR difficult without significant manual fine-tuning. This hinders wider adoption of HLS.

We advocate for *open EDA verticals* as a solution to enabling more widespread use of domain-specific hardware acceleration. The objective is to empower small teams of domain experts to productively develop high-performance accelerators using programming interfaces they are already familiar with. For example, this means supporting domain-specific frameworks like PyTorch or TensorFlow for ML applications. In order for EDA verticals to proliferate, there must first be extensible infrastructure similar to LLVM [5] and MLIR [**?**] from which to build new tool flows. The proper EDA infrastructure would include novel intermediate representations specifically tailored to the unique challenges in gradually lowering high-level code down to gates.

The CIRCT project [2] is the closest work that exists to a true inter-operable hardware design infrastructure made for open EDA verticals. In fact, CIRCT was chartered specifically for this purpose. Building on the MLIR project, CIRCT defines various dialects that are relevant to hardware design, such as RTL, FSM, handshake, and pipeline. These dialects provide a standardized way for exchanging information between tools at different levels of the compilation process, both within and across layers. The key benefit of this approach is that the dialects are specifically designed to meet the needs of downstream compilers, as opposed to the conventional practice of using Verilog or VHDL as the representation for interchanging or signing-off designs between EDA tools.

The primary focus of CIRCT so far has been simplifying RTL compilers. While it provides some support for high-level design specification, implementing a traditional HLS tool in CIRCT does not address the underlying problem HLS faces. Better EDA verticals hinge on programming models that can (1) bridge the gaps between popular domains, (2) provide designers with greater control on important customizations, (3) and serve as a compilation target for multiple high-level domain-specific languages (DSLs). There are several ongoing efforts that attempt to tackle this challenge. For example, Calyx [9] is an intermediate representation that captures both the temporal control flow of programs and reusable hardware resources, making it easier to build compilers for domain-specific languages that generate RTL. HeteroCL [**?**] provides a Python-based intermediate language that separates the specification of algorithms from hardware customization in terms of data types, compute, and memory architectures; it further raises the abstraction level of accelerator design and can serve as a focal point for integrating various HLS optimizations and supporting high-level DSLs.

To reiterate, the newest experimental EDA tools aspire to better exploit domain-specific information to simplify optimizations and provide better upfront QoR with minimal intervention from engineers. Nevertheless, there are still many other unanswered questions related to open EDA verticals for physical design, and these questions present great research opportunities for the physical design community to explore. This leads into the second component of EDA verticals: a vertically integrated approach allows for

ISPD '23, March 26–29, 2023, Virtual Event, USA

ACM ISBN 978-1-4503-9978-4/23/03.
https://doi.org/10.1145/3569052.3578905

cross-stack optimizations and for them to take place at higher levels of abstraction. By sharing more context across the stack, each tool in the flow can have a tighter focus on its specific role. As a consequence, redundant steps are eliminated, and there is a reduced chance that stages of the flow will optimize in opposition to one another. As for reducing the cost of creating new verticals, it will be important to use open standard interfaces as interchange between different tools so as to capture the commonality of compiling for different domains.

Several recent efforts have demonstrated the benefits of using higher-level context to improve physical design. For example, AutoBridge [**?**] proposes a solution for the frequency degradation of designs spanning multi-die FPGAs by performing coarse-grained floor-planning of the HLS functions. AutoBridge uses the dataflow graph to distribute the HLS functions across the FPGA dies and insert additional pipeline registers in RTL. This work demonstrates cases where the ordinary register re-timing is not sufficient, and it is a good example of the higher-level optimizations awarded by vertically integrated flows. In this case, restricting the input HLS designs to a dataflow model enables a close coupling of placement and the HLS frontend, and this lessens the difficulty of the compilation for the backend. However, AutoBridge achieves these results through complex modification of the RTL produced by HLS. Even one step lower, one could imagine having an interface to directly modify and place netlists. As a follow-up work to AutoBridge, RapidStream [**?**] also partitions designs at the HLS level. However, this work compiles the partitions in parallel and bridges the separately compiled pieces by directly editing the netlists and partial placements. The higher-level knowledge about how partitions will be laid out on the device allows for a simple router in the stitching phase that is much faster than a full-featured general-purpose router. RapidStream achieves this through an open-source framework, called RapidWright [6], which enables custom place-and-route algorithms for AMD Xilinx FPGAs. In the end, the work was able to achieve both compile time speedups and increases in clock frequency. On one level, open EDA verticals would encourage more tools like RapidStream and RapidWright to exist. More importantly, having open standard interfaces would allow these tools to be used in conjunction for stacking benefits.

As the engineering effort and cost required to build accelerators with conventional EDA flows continue to increase, we need to seek out productivity increases through domain-specific approaches. Tighter vertical integration in these new EDA flows will allow productivity and QoR gains that have been impractical to attain in the past. Moreover, open standard interfaces will enable the development of better domain-specific tools with less effort by seamlessly reusing parts of existing tools. Finally, using open standards in EDA verticals will naturally encourage the use of more fully open-source tools. In the end, domain-specific flows, with their cross-stack optimizations and improved design productivity, will be preferred to manual general-purpose flows and attract new customers of hardware acceleration. Overall, we believe that open EDA verticals are crucial for the future of the semiconductor industry. To fully realize its potential, it will require close collaboration from hardware engineers, compiler and EDA tool developers, and domain experts across various fields.

ACM Reference Format:
Zhiru Zhang, Matthew Hofmann, and Andrew Butt. 2023. A Case for Open EDA Verticals. In *Proceedings of the 2023 International Symposium on Physical Design (ISPD '23), March 26–29, 2023, Virtual Event, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3569052.3578905

SPEAKER BIOGRAPHY

Zhiru Zhang is an Associate Professor in the School of ECE at Cornell University. He is an IEEE Fellow. His current research investigates new algorithms, design methodologies, and automation tools for heterogeneous computing. His research has been recognized with a Facebook Research Award, Google Faculty Research Award, the DAC Under-40 Innovators Award, the Rising Professional Achievement Award from the UCLA Henry Samueli School of Engineering and Applied Science, a DARPA Young Faculty Award, and the IEEE CEDA Ernest S. Kuh Early Career Award, an NSF CAREER Award, the Ross Freeman Award for Technical Innovation from Xilinx, and multiple best paper awards and nominations. Prior to joining Cornell, he was a co-founder of AutoESL, a high-level synthesis start-up later acquired by Xilinx (now AMD).

REFERENCES

[1] T. Ajayi, V. Chhabria, M. Fogaça, S. Hashemi, A. Hosny, A. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem, G. Pradipta, S. Reda, M. Saligane, S. Sapatnekar, C. Sechen, M. Shalan, W. Swartz, L. Wang, Z. Wang, M. Woo, and B. Xu. 2019. Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project. *Design Automation Conference (DAC)* (2019).
[2] CIRCT 2020. *CIRCT Charter.* Retrieved January 28, 2023 from https://circt.llvm.org/docs/Charter/
[3] Florent Kermarrec, Sébastien Bourdeauducq, Jean-Christophe Le Lann, and Hannah Badier. 2020. "LiteX: an open-source SoC builder and library based on Migen Python DSL". (2020). https://doi.org/10.48550/ARXIV.2005.02506
[4] KLayout 2006. *KLayout.* Retrieved January 28, 2023 from https://www.klayout.de
[5] C. Lattner and V. Adve. 2004. LLVM: A compilation framework for lifelong program analysis and transformation. *Int'l Symp. on Code Generation and Optimization* (2004).
[6] C. Lavin and A. Kaviani. 2018. "RapidWright: Enabling Custom Crafted Implementations for FPGAs". *Int'l Symp. on Field-Programmable Custom Computing Machines (FCCM)* (2018).
[7] K. Murray, M. Elgammal, V. Betz, T. Ansell, K. Rothman, and A. Comodi. 2020. "SymbiFlow and VPR: An Open-Source Design Flow for Commercial and Novel FPGAs". *IEEE Micro* (2020).
[8] K. E. Murray, O. Petelin, S. Zhong, J. M. Wang, M. ElDafrawy, J.-P. Legault, E. Sha, A. G. Graham, J. Wu, M. J. P. Walker, H. Zeng, P. Patros, J. Luu, K. B. Kent, and V. Betz. 2020. "VTR 8: High Performance CAD and Customizable FPGA Architecture Modelling". *ACM Transactions on Reconfigurable Technology and Systems (TRETS)* (2020).
[9] R. Nigam, S. Thomas, Z. Li, and A. Sampson. 2021. "A Compiler Infrastructure for Accelerator Generators". *Int'l Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)* (2021).