2025 IEEE 33rd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM) | 979-8-3315-0281-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/FCCM62733.2025.00050

RapidPnR: Accelerating the Physical Design for FPGAs via Design-Level Parallelism

Wanzheng Weng, Pingqiang Zhou
School of Information Science and Technology, ShanghaiTech University, Shanghai, China
Email: {wengwzh2022, zhoupq}@shanghaitech.edu.cn

***Abstract*—The runtime of physical design has become a critical issue for FPGA development as the scale and complexity of circuit designs surge with the increasing logic capacity of FPGA devices. The time-consuming process of physical design significantly extends the cycle of design iteration, which heavily impacts the efficiency of debugging and architecture optimization. To address this issue, this work proposes a generic, fully-automated and split-and-parallel physical design flow to accelerate the deployment of large-scale circuits on FPGAs. Specifically, our flow automatically partitions the synthesized netlist into multiple smaller pieces, performs parallel physical design of each piece, and then merges them into the complete design. Evaluated on a set of real circuit benchmarks, our flow reduces the runtime by more than 50% and ensures nearly the same design frequency compared to the physical design flow provided by the commercial tool Vivado.**

As the scale and complexity of circuits continue to grow, the runtime issue of physical design on FPGAs has become increasingly significant. The time-consuming process of placement and routing directly extends the cycle of design iteration, which affects the efficiency of debugging and performance optimization of target designs.

In recent years, numerous efforts have been made to accelerate the physical design for FPGAs, and their methodologies can generally be classified into two categories: **Algorithm-level parallelism** first decomposes key physical design algorithms, such as placement [3] and routing [4], into multiple independent tasks, then executes these tasks in parallel on GPUs [3] or multicore CPUs [4]. Most of these works don't consider timing optimization, which accounts for most runtime of high-performance physical design and is difficult to parallelize [1]. **Design-level parallelism** [1], [2] directly partitions the original circuit into multiple disjoint islands, then performs physical design on each island concurrently, and finally merges them into the complete design. The state-of-the-art design-level parallelism works are only applicable to HLS designs with the dataflow architecture. Their implementation relies on the module hierarchy information provided by the design language and latency-insensitive nature of dataflow designs.

In this work, we explore the idea of design-level parallelism at the netlist level and propose a fully automated, split-and-parallel and high-performance physical design flow which is independent of detailed circuit architectures or front-end design languages. More specifically, our flow consists of three main stages as illustrated in Fig. 1. The first stage transforms the synthesized netlist into an abstract hypergraph, which aims to conceal some critical nets that may cause routing errors or timing violations if cut during partitioning. In the second stage, the entire FPGA is divided into disjoint islands using a grid. Then we develop an efficient two-stage algorithm to distribute abstract vertices across these islands, ensuring multiple utilization constraints and minimizing the number of inter-island nets. To guide the placement and routing of inter-island nets, we extract the source cell of each inter-island net and assign it to the boundary region. Then we perform physical design of each island along with its boundary regions in parallel and merge them into the complete design.

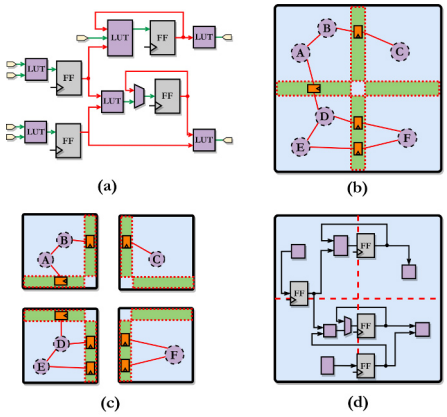

Fig. 1. An illustration of proposed split-and-parallel physical design flow: (a) the synthesized netlist; (b) the abstract hypergraph and its partition and floorplan results; (c) split and parallel physical design of each island; (d) the complete design after merging islands together.

Evaluated on a set of FPGA designs with diverse architecture patterns, our flow speeds up the process of physical design by 1.6x–2.5x with an average frequency degradation of only 6% compared to the commercial tool Vivado.

This work was financially supported by the Science and Technology Commission of Shanghai Municipality (STCSM) under Grant 24JD1402500.

REFERENCES

[1] L. Guo, et al. "Rapidstream: parallel physical implementation of fpga hls designs," in Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, pp. 1–12, 2022.
[2] Y. Xiao, et al."Fast Linking of Separately-Compiled FPGA Blocks without a NoC," International Conference on Field-Programmable Technology, Maui, HI, USA, 2020, pp. 196-205.
[3] J. Mai, et al. "Multi-electrostatic fpga placement considering slicel-slicem heterogeneity and clock feasibility," in Proceedings of the 59th ACM/IEEE Design Automation Conference, pp. 649–654, 2022.
[4] T. Martin, et al. "A High-Performance Routing Engine for Large-Scale FPGAs," International Conference on Field-Programmable Logic and Applications, Torino, Italy, 2024, pp. 53-59.

2576-2621/25/$31.00 ©2025 IEEE
DOI 10.1109/FCCM62733.2025.00050