Technical Paper Proposal: Placement Algorithms for Heterogenous FPGAs

Brian B Cheng
Department of Electrical and Computer Engineering

1 Keywords

- FPGA, EDA, Synthesis, Placement, Routing, Parallel, Optimization

2 Proposal

In modern software development, the process of compiling high-level languages like C++ or Java into executable machine code is highly efficient and streamlined. The software toolchains only consist of compilers and scripts and are mostly open source and easy to learn. The compilation is where most of the magic happens. Through multiple stages of lexing, parsing, and optimizations, the compiler takes the developer's program and produces working machine code. The developer can influence a few optimization flags if needed, but for the most part, the developer simply runs the default compiler command and is presented with the final product. All of this happens within milliseconds to seconds, even for large projects.

In the same way that a compiler takes a high-level C++ file and assembles machine code ready to be executed on a CPU, an Electronic Design Automation (EDA) tool takes a high-level Verilog file and produces a hardware configuration in the form of a bitstream, ready to be deployed onto an FPGA or ASIC. In a superficial way, they perform the same task but for different industries. One for software, the other for hardware.

In the FPGA world, the design entry to product pipeline is despairingly complex by comparison. First, the design entry. The engineer describes a digital system using a high-level HDL like Verilog or VHDL. Then the engineer submits the design entry to an EDA like Vivado or Quartus to perform three stages of automated design. These are: synthesis, placement, and routing. All three are NP-hard problems. For even modest designs like binary counters or 8-bit CPUs that utilize a small percentage of the available resources on the FPGA, the EDA will spend a minimum of a couple minutes synthesizing and implementing the design. The EDA run time increases exponentially with the scale of the project. Projects that utilize 80% or more of the FPGA's resources may run for hours, and for the high-end devices like Xilinx's Kintex FPGAs

which have millions of logical elements, can run for days. At high utilization, sometimes the design EDA cannot fit the design onto the device due to routing congestion even when it has enough logical elements to implement the design, essentially wasting hours or days of runtime.

One of the most common complaints from those in the FPGA design industry, especially from those in software development, is that the FPGA toolchains are overall slow and heavy. Unlike in software development where an error in the program can be patched and recompiled in a matter of minutes, EDA runtimes lasting several hours means that an engineer can only go through a couple design cycles per day. A Verilog developer must be very precise with their coding and practice thorough verification of their design before submitting it to the EDA. This overall heightens the barrier of entry into FPGA development and contributes to a shortage of qualified FPGA engineers.

This paper aims to attack one of the sections of this wall: the placement stage. Like synthesis and routing, it is an NP-hard problem. The search space for a placed design is massive.

3 Abbreviations

- **FPGA**: Field Programmable Gate Array
- **VLSI**: Very Large Scale Integration
- **EDA**: Electronic Design Automation
- **VHSIC**: Very High Speed Integrated Circuits
- **HDL**: Hardware Description Language
- **VHDL**: VHSIC HDL
- **HLS**: High Level Synthesis: Generating synthesizable HDL from high-level software languages. A company might want to have a software engineer write C or C++ code and have a program translate it into synthesizable Verilog. This can can boost productivity and save the company the need to hire a hardware engineer.
- **IP**: Intellectual Property: In FPGA context, this means pre-built modules or subsystems like a hardened microprocessor or Ethernet controller. These are usually proprietary.
- **SoC**: System on Chip: An FPGA device (chip) that features hardened IP in addition to the programmable logic fabric.
- **PL-PS**: Programmable Logic - Processing System: A design that utilizes the on-chip hard microprocessor in conjunction with the programmable logic fabric.
- **EDIF**: Electronic Design Interchange Format

- **HPWL**: Half Perimeter Wire Length

4 Ideas

- **FPGA**: Field Programmable Gate Array
  - FPGA Vendors:
    - AMD-Xilinx (∼50% FPGA vendor market share)
    - Intel-Altera (∼35% share)
    - Lattice
    - Microsemi
- **EDA**: Electronic Design Automation
  - Proprietary software for FPGA and VLSI development:
    - Xilinx - Vivado (Design + Simulation) + Vitis (HLS + PL-PS codesign)
    - Altera - Quartus (Design) + ModelSim (Simulation)
    - Synopsis (VLSI)
    - Cadence (VLSI)
  - Open source software for FPGA development:
    - **VTR**: Simulated Annealing placer for FPGAs. Popular among researchers who study placement techniques. Commonly referred to as an "academic placer".
    - **OSS-CAD**: a full-flow software suite that includes ABC synthesis, Yosys synthesis, Yosys nextpnr.
    - **AMF-Placer**: Analytical Placer for FPGAs
    - **RapidWright**: Semi-open source API that provides backend access to Xilinx Vivado EDA using design checkpoints.
    - **RapidLayout**: Hard Block Placer for Systolic Arrays. Built with RapidWright.
    - **RapidStream**: HLS Placer. Built with RapidWright.
    - **DREAMPlace**: GPU-powered deep learning placement for VLSI.
    - **DREAMPlaceFPGA**: DREAMPlace, adapted to FPGAs via the RapidWright API.
- **Synthesis**
  - Takes a design written in a high-level HDL like VHDL or Verilog and "synthesizes" a **logical netlist** out of it.
  - The logical netlist is usually generated as an EDIF, JSON, or a low-level Verilog file.

  - The netlist describes the necessary basic elements of logic (BELs) and the wired connections between them that are necessary to implement the design.

- **Placement**
  - Takes the **logical netlist** and produces a **physical netlist**.
  - For each BEL in the netlist, assign the BEL to a Cell, Site, and Tile on the physical FPGA device.

- **Routing**
  - Takes the **physical netlist** and maps the connections between BELs onto wires, interconnects, and switchboxes on the FPGA.

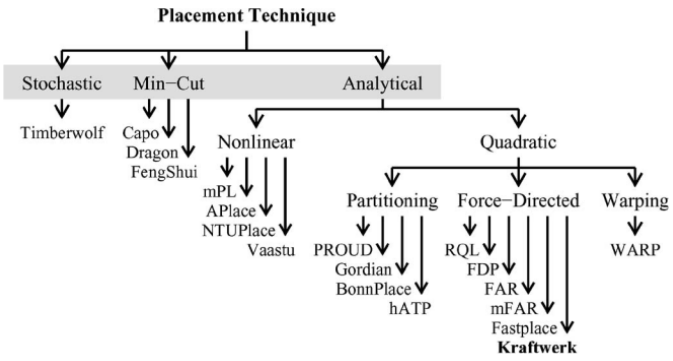

Figure 1: Landscape of VLSI placement techniques (Spindler) [1]

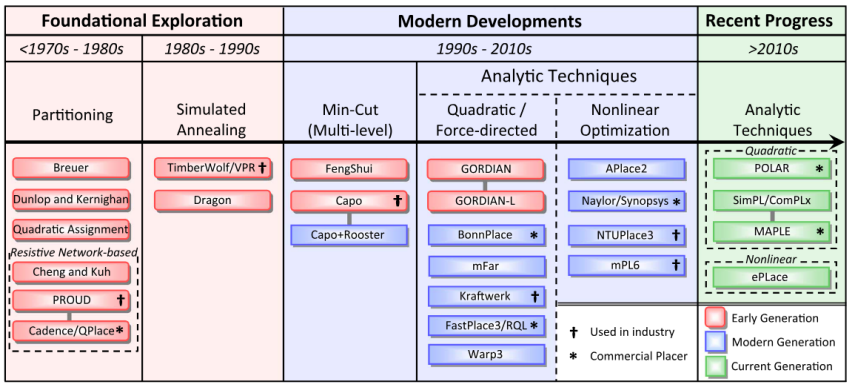

Figure 2: Historical timeline of VLSI placement techniques (Markov) [2]

This is a citation for AMFPlacer. [3]

References

[1] P. Spindler, U. Schlichtmann, and F. M. Johannes, "Kraftwerk2—a fast force-directed quadratic placement approach using an accurate net model," *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 27, p. 1398–1411, aug 2008.

[2] I. L. Markov, J. Hu, and M.-C. Kim, "Progress and challenges in vlsi placement research," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 1985–2003, 2015.

[3] T. Liang, G. Chen, J. Zhao, S. Sinha, and W. Zhang, "Amf-placer: High-performance analytical mixed-size placer for fpga," in *2021 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–6, 2021.

[4] A. Mishra, N. Rao, G. Gore, and X. Tang, "Architectural exploration of heterogeneous fpgas for performance enhancement of ml benchmarks," in *2023 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 232–235, 2023.

[5] U. Farooq, H. Parvez, H. Mehrez, and Z. Marrakchi, "Exploration of heterogeneous fpga architectures," *Int. J. Reconfig. Comput.*, vol. 2011, jan 2011.

[6] C. Lavin, M. Padilla, S. Ghosh, B. Nelson, B. Hutchings, and M. Wirthlin, "Using hard macros to reduce fpga compilation time," in *2010 International Conference on Field Programmable Logic and Applications*, pp. 438–441, 2010.

[7] C. Lavin and A. Kaviani, "Rapidwright: Enabling custom crafted implementations for fpgas," in *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 133–140, 2018.

[8] Y. Zhou, P. Maidee, C. Lavin, A. Kaviani, and D. Stroobandt, "Rwroute: An open-source timing-driven router for commercial fpgas," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 15, nov 2021.

[9] C. Lavin and E. Hung, "Invited paper: Rapidwright: Unleashing the full power of fpga technology with domain-specific tooling," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, pp. 1–7, 2023.

[10] A. A. Kennings and I. L. Markov, "Analytical minimization of half-perimeter wirelength," in *Proceedings of the 2000 Asia and South Pacific Design Automation Conference*, ASP-DAC '00, (New York, NY, USA), p. 179–184, Association for Computing Machinery, 2000.

[11] S. Areibi, G. Grewal, D. Banerji, and P. Du, "Hierarchical fpga placement," *Canadian Journal of Electrical and Computer Engineering*, vol. 32, no. 1, pp. 53–64, 2007.

[12] M. Gort and J. H. Anderson, "Analytical placement for heterogeneous fpgas," in *22nd International Conference on Field Programmable Logic and Applications (FPL)*, pp. 143–150, 2012.

[13] J. Chen, W. Zhu, J. Yu, L. He, and Y.-W. Chang, "Analytical placement with 3d poisson's equation and admm based optimization for large-scale 2.5d heterogeneous fpgas," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, 2019.

[14] S. Dhar, L. Singhal, M. Iyer, and D. Pan, "Fpga accelerated fpga placement," in *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*, pp. 404–410, 2019.

[15] R. S. Rajarathnam, M. B. Alawieh, Z. Jiang, M. Iyer, and D. Z. Pan, "Dreamplacefpga: An open-source analytical placer for large scale heterogeneous fpgas using deep-learning toolkit," in *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 300–306, 2022.

[16] H. Long, Y. Bai, Y. Li, J. Wang, and J. Lai, "Optimizing wirelength and delay of fpga tile through floorplanning based on simulated annealing algorithm," in *2023 IEEE 15th International Conference on ASIC (ASICON)*, pp. 1–4, 2023.

[17] T. Liang, G. Chen, J. Zhao, S. Sinha, and W. Zhang, "Amf-placer 2.0: Open source timing-driven analytical mixed-size placer for large-scale heterogeneous fpga," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2024.

[18] B. Ray, A. R. Tripathy, P. Samal, M. Das, and P. Mallik, "Half-perimeter wirelength model for vlsi analytical placement," in *2014 International Conference on Information Technology*, pp. 287–292, 2014.

[19] B. N. B. Ray, S. K. Mohanty, D. Sethy, and R. B. Ray, "Hpwl formulation for analytical placement using gaussian error function," in *2017 International Conference on Information Technology (ICIT)*, pp. 56–61, 2017.

[20] Y. Lin, S. Dhar, W. Li, H. Ren, B. Khailany, and D. Z. Pan, "Dreampiace: Deep learning toolkit-enabled gpu acceleration for modern vlsi placement," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2019.