Integrated Circuit Design Using Open-Source Tools

Dylan Grace, Jeremy Blumka, Joel Reuning-Scherer, Larissa Ptak, Philip Wig, Adviser: Dan White, Ph.D.

Abstract

Our team is exploring open-source tools for Very Large-Scale Integration (VLSI) design, used in the design of computer chips. In the past, most tools for chip design have been proprietary. We are exploring using open-source tools for both analog and digital VLSI design. By furthering our understanding of these tools, we aim to make VLSI design accessible to a broader market of people, in order to aid in education by removing the barriers to learning about integrated circuit design. This will allow for not just theoretical education but also practical experience in the chip design field for undergraduate students.

The Importance of Approachability

In order to produce a functional chip, one needs a sound understanding of architecture design, basic logic design, logic verification, physical design layout, physical design verification, fabrication, and testing. The wider availability of an open-source design flow ultimately will produce more competent and capable engineers with more diverse backgrounds. By testing and documenting our process with an open-source design flow, we are increasing awareness and availability of the chip design process.

The Digital Design Process

Most modern devices are digital systems due to their deterministic behavior. The digital design process is far more streamlined than its analog counterpart, as software tools exist to implement hardware behavior via standard cells. Due to the simplicity in the digital design process, digital systems can be much more complex than analog systems.

The digital design process can be generalized as follows:

- Define system requirements/constraints, such as minimum clock speed, target power consumption, or desired inputs and outputs
- Determine the system architecture based on design constraints
- Implement the system in a hardware description language (Verilog)
- Simulate and validate input and output waveforms (Icarus Verilog, GTKWave, Ngspice, Quartus ModelSim)
- Procedurally generate a floorplan of the system (OpenLane) using standardized cells (SkyWater PDK, Efabless Caravel)
- Verify and finalize the physical layout of the design for tape-out (OpenLane, KLayout, Magic, Ngspice), including integration with Caravel

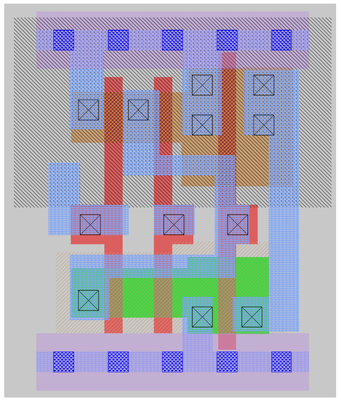

Figure 1 (above): *And2 standard cell from SkyWater PDK* [1].

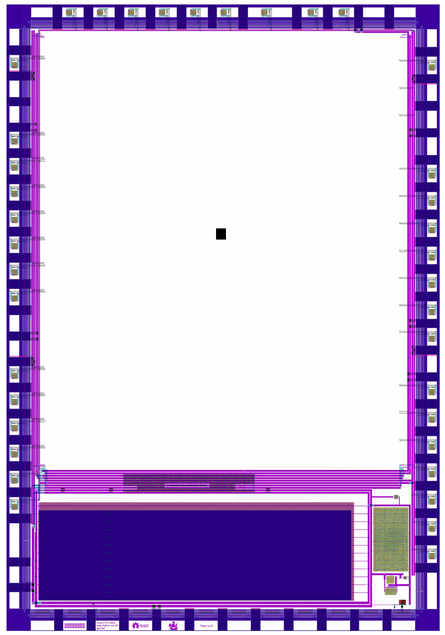

Figure 2 (right): *The Efabless Caravel harness. Actual size is 3.6mm x 5.2mm. The entirety of the ALU depicted in Figure 3 fits within the black region in the center* [2].

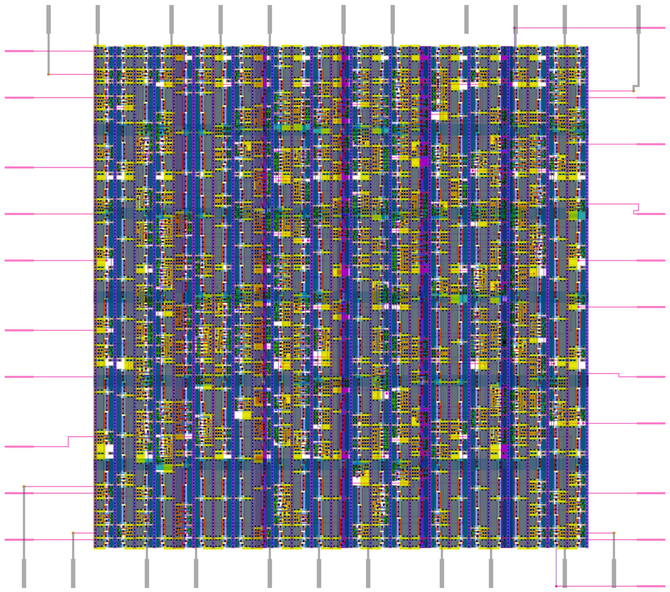

Figure 3: *Custom Arithmetic Logic Unit (ALU) made with OpenLane. This custom ALU is 80.685 microns wide and 91.405 microns tall.*

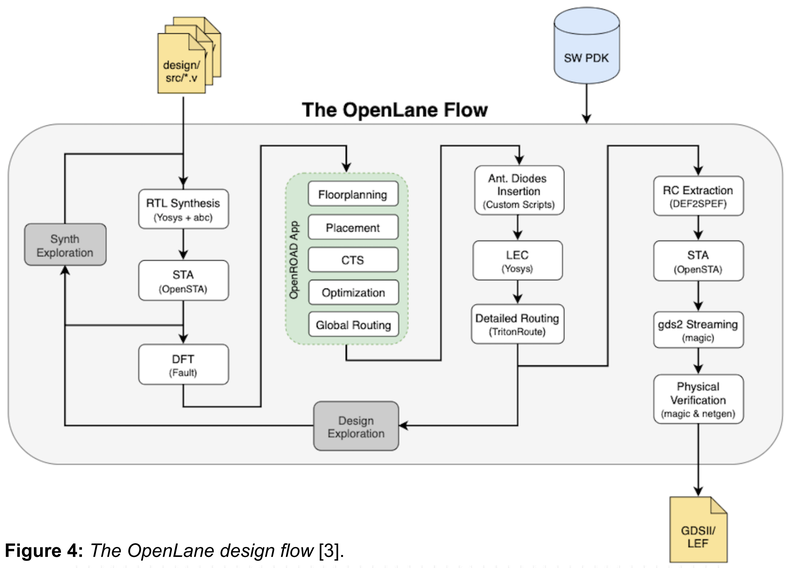

Figure 4: *The OpenLane design flow* [3].

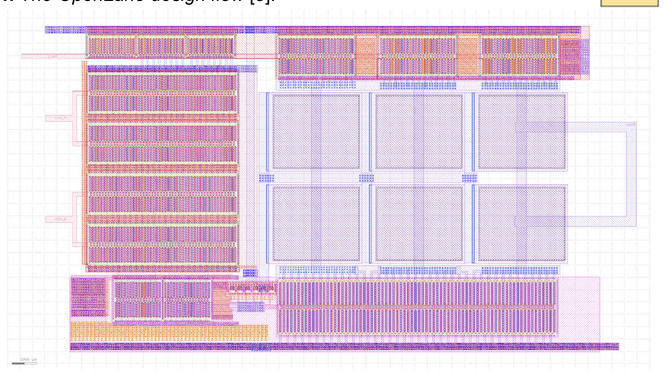

Figure 5*: Layout for a two stage Miller compensated operational amplifier using KLayout* [4].

The Analog Design Process

Unlike the highly automated digital flow, analog design using the SkyWater PDK is a manual process. Within an analog design, each transistor needs to be individually sized and designed.

The analog design process can be generalized as follows:

- Define system requirements/constraints, such as power consumption, accuracy tolerances, etc.
- Determine the system architecture, based on performance constraints
- Specify transistor characteristics for each transistor in the design
- Simulate the design to verify functionality
- Finalize the physical layout of the design for tape-out

For our designs, Xschem was used for schematic design, Ngspice was used for simulations, and Magic or KLayout was used for the physical layout. Design Rule Checking (DRC) was performed in Magic and Layout Versus Schematic (LVS) checking was done using Netgen.

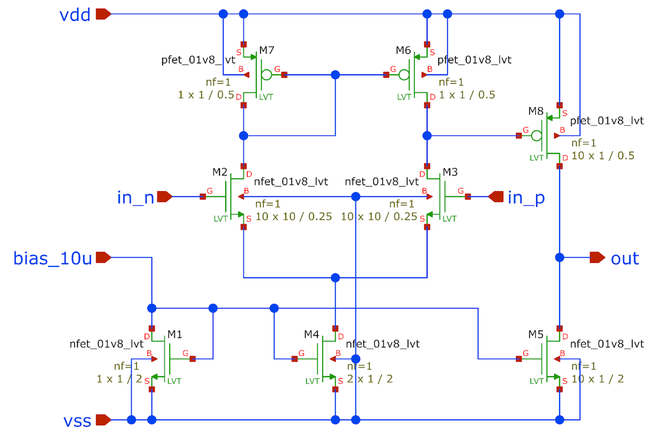

Figure 6: *Two-stage operational amplifier, designed in Xschem.*

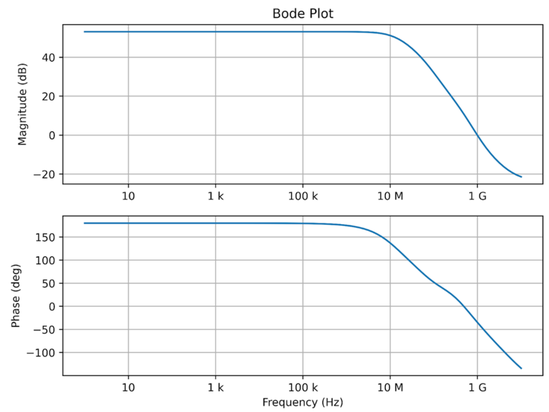

Figure 7: *Frequency response plot of the operational amplifier shown in Figure 6, simulated with Ngspice and plotted using Python and the Matplotlib library.*

References:

[1] Google. (2022). SkyWater Open Source PDK. GitHub Repository. Available: https://github.com/google/skywater-pdk
[2] Efabless. (2022). Caravel Harness. GitHub Repository. Available: https://github.com/efabless/caravel
[3] Ghazy, A., & Shalan, M. (2020). OpenLANE: The open-source digital ASIC implementation flow. Presented at Workshop on Open-Source EDA Technol. (WOSET). [PDF]. Available: https://github.com/woset-workshop/woset-workshop.github.io/blob/f2171edef70cf2fb5a87666554873a6a77eaa2c6/PDFs/2020/a21.pdf
[4] Hernando, D. (2021). Caravel_fulgor_opamp. GitHub Repository. Available: https://github.com/diegohernando/caravel_fulgor_opamp

WIRED
Wireless Research and Electronic Discovery

VALPARAISO UNIVERSITY
ENGINEERING