Free IC Design in Education

Aleksandar Pajkanovic

Faculty of Electrical Engineering, University of Banja Luka, Patre 5 78000 Banja Luka,
The Serbian Republic, Bosnia and Herzegovina
e-mail: aleksandar.pajkanovic@etf.unibl.org

Abstract

One of the most important meetings that never happened was when King Theoden led his Riders of the Mark into a magnificent charge, releasing Minas Tirith from the siege, joining forces with King Aragorn of Gondor and winning the Battle of Pelennor Fields. About as impossible as that, not so many years ago, was the meeting of free software and open hardware at the nanoelectronics scale, down at the IC design level. Believe it or not, thanks to the many hours of many hard working engineers across the globe we are now able to design chips using nothing but the free resources: tools, PDK and IP alike. In this paper, we present our small contribution, mirrored in organizing analog and digital IC design courses at the Faculty of Electrical Engineering in Banja Luka, using the free and open source material.

Key words: CMOS, IC, chip design, analog, digital, EDA tools, free software, open hardware

1 Introduction

CMOS integrated circuit (IC) fabrication is the most sophisticated technology process commercially available on Earth. Therefore, its significance in the education of the next and, for that matter, every generation of electronics engineers is of paramount importance. Hence, it is the duty of educators in this domain around the world to bring IC courses to the universities in a manner that students actually get silicon proven experience. However, due to required funding and the complexity of legal paperwork that is not an easy task.

In the last year or so, a revolution in this domain is taking place, enabling us to do exactly that - have our students fabricate IC on silicon by removing both obstacles: funding and legal issues. This is done through the evolution of free electronic design automation (EDA) tools and open-sourcing a project design kit (PDK) In this paper we share some experiences of our own on organizing both analog and digital CMOS IC design courses at the University of Banja Luka. A retrospective overview of how we developed these programmes may be found in [1]–[5], by phases.

In section 2, we provide insight on why do we actually need the EDA tools and the PDKs by taking a look at what did take to create chips without these. Then, in section 3, we explain the shortcomings of the mainstream, industry standards tools and PDKs. In section 4, we describe the possibility of free and open ICs at all abstraction levels, as in free of charge and open to sharing with anyone every detail of the design and technology procedures. In sections 5 and 6, we present how analog and digital IC design courses are organized at the University of Banja Luka, respectively, utilizing the revolution at hand. Finally, a conclusion follows.

2 IC design, no tools, no PDK

Back when it all started, half a century ago and a bit more, with engineering legends breaking the ice in bipolar and MOSFET IC design, in the 60s and the 70s - there were no problems with IC design tools,

neither PDKs, nor intellectual property (IP) access; there were no problems simply because neither of those three components actually existed. Today, however, either of these three represents a sine qua non - and each one brings great benefits. To understand how they make IC designer life easier, let's take a moment to consider a testimonial from Y. Tsividis [6]:

> [..] to make analog circuits work in an unproven technology for such applications, we had to invent new schemes to circumvent the technology's limitations. [...] but there were no such techniques for analog MOS integrated circuits at that time. [...] there were no silicon foundries. The concept did not even exist as an idea at the time. So to verify our designs, we had to make the chips ourselves in the lab [...] Circuits were laid out manually, using a back-breaking process involving a light table, on top of which we would lay flat a "rubylith" sheet [...] With proper lighting, we would then photograph the result with a large reduction factor, and then photograph the result again with another reduction factor, to produce the final glass plate that was used as a mask in fabrication.

All that, while keeping in mind that the Integrated Circuits Lab (the "lab" mentioned in the quoted paragraph above), was actually privileged with access to SPICE, being developed at the time right next door, also at UC Berkeley. That said, let's assume there was a sort of a simulator (taking input with punchcards!), but drawing layout manually would still mean the IC designer would have to stand around a table a few meters across with (a sort of) pen and knife to draw both devices and interconnects, i.e. place and route". Then, that very same engineer, the IC designer, would have to take a chance at dealing with poisonous chemicals and matter at dangerously high temperatures, to get the design into a chip, i.e. to "harden" the RTL/schematic. The probability of all that working out is quite low, so many a try would have to be done for the very same project to actually see it operational at the end. And not only the amplifier.

Thanks to the development of the electronic design automation (EDA) tools (and, obviously, all other prerequisites) and appearance of the silicon foundry (fab, for short), instead of asking themselves: "Which knife is the sharpest?", or "Which fume will intoxicate me?", IC designers ask themselves things like: "Which chair is more comfortable?", or "Should I add a third monitor?"

3 IC design, no free tools, no free PDK

Very soon after the IC market exploded, the importance of simulators and, consequently, other tools was realized. Beside the semiconductor industry and all the consumer, industrial and military electronics markets dependent on it, that realization also created the industry of EDA tools specified for IC design.

Over the next several decades, the struggle to create a more efficient and a more effective tool, capable of following the Moore's law, i.e. processing the ever decreasing features, today well into one-digit nanometer numbers, took place. As a result, industry standard EDA tools in this domain are few and all are provided by large companies, well established vendors: Cadence, Synopsis and Mentor. All of them do provide discounts for both Higher Education institutions and startups, under certain limitations. Nevertheless, if one is to use these in a commercial environment, a significant budget per seat on a yearly basis must be assigned to a commercial license, making it beyond the reach of many.

The free tools have been around since the 80s, sure. However, due to mainly two reasons, these tools very soon became unusable in practice. First, it was not an easy job supporting newer and newer technology nodes, i.e. following the scaling dictated by Moore's law. Second reason requires a bit of context. To get the idea, we need to point out that, by this time (80s and 90s), silicon foundries were a common place, as well as circuit simulators and layout software tools, etc. Therefore, the process

described by prof. Tsividis has been made both more efficient and more effective. Primarily, efficiency came from the increased precision of device models used for simulation and effectiveness came from the professionals working in foundries, where ICs were made in larger batches. The Project Design Kit (PDK) is the glue that makes the simulation results resemble the silicon fabricated IC characterization in the lab. PDK of a process node holds device models fine tuned to the exact technology process node that is under way within the foundry. Hence, the IC designer may trust the results obtained through simulation, as it is highly probable that the circuit in silicon will yield the same behavior. Of course, for the EDA tool to be useful, it must be compatible with a PDK. Obviously, the information collected within a PDK is valuable. Very valuable. Hence, it is always subject to at least NDAs and sometimes quite limiting terms and conditions of use. In other words, even if there was the best EDA tool ever, unless a foundry decides they have interest in supporting the tool by providing a PDK compatible with that tool - the tool is useless. Now, the second reason why the free EDA tools haven't been in widespread use comes from the fact that most foundries that carried semiconductor's progress through node scaling haven't seen a reason to support free tools. It is worth mentioning the scalable CMOS (SCMOS), created by MOSIS, but this is limited to universities in the US and very, very mature nodes. Europractice allows multiproject wafers on this side of the ocean, they offer discounted prices and an abundance of PDKs - yet, neither with support for the free EDA tools.

Finally, let's assume that the free tools are usable enough to the point that we actually do a design in a very old technology node - these nodes do have educational value, after all. We can do that through the MOSIS service, or something similar. That design, it's layout and other technology-specific details are still subject to the NDA issued by the fab, further meaning the design may not be treated as open hardware. This limits it's educational potential heavily and, to conclude, renders the free tools - without a free PDK - not free tools (even though they do come with a license that is free of charge), since we are unable to make free hardware.

4 IC design, free tools, free PDK

The work on the free tools, taking them to the point to actually deliver a silicon proven design, has intensified about a decade ago, when the company efabless [7] had taken up the struggle. Leveraging plenty of tools from Open Circuit Design [8], supported by Tim Edwards for years and some other famous open-source tools, primarily ngspice [9], they created an online platform that designers may access remotely. The PDK issue was resolved by including technology nodes from XFAB in a manner that obscures the proprietary data. In this way, they enabled anyone to log in and use the tools and technology without signing the NDA personally and without fiddling with the tools configuration and installation. Still, the designs created in this way were no free open hardware. Nevertheless, they've proven that it is possible to create silicon ready designs using nothing but the free and open source tools [10]. More people joined forces producing further evidence that this is an important effort, the most prominent project being ASICOne [11]. We were fortunate enough to take part as well, and, among the first in the world, produce silicon ready designs using that methodology within an undergraduate course[3].

As emphasized in the previous section, all this effort invested in the tool development was still relaying on proprietary PDKs. Hence the revolution of open hardware at ASIC level started only after the missing link has been introduced in June 2020, by Tim Ansell, representing the Google efforts [12] in providing the first ever open source PDK of a CMOS technology node. Even though the 130 nm process appeared on the market back in 2001, it is still used in the area of research, small microcontroller development, and mixed signal embedded designs, i.e. IoT devices. Skywater Technology [13] was founded recently, through acquisition of Cypress' subsidiary, a foundry with extensive experience in IC fabrication. Therefore, the Skywater 130 nm (SKY130 [14]) process node, while open-sourced in 2020, is derived from a mature CMOS technology. On the other hand, while that original CMOS

technology has been used in design of many successfully manufactured IC in commercial purposes, the SKY130 PDK is not yet production ready. It is currently intended only for making test chips and prototype verification. Still, it is important to note that neither of those is guaranteed, at this point - i.e. this PDK is still a work in progress.

SKY130 is a 5 metal levels 180 nm-130 nm hybrid process node with internal input/output pins operable at both 1.8 V and 5 V. It includes MiM capacitors, inductors, one level of local interconnect and SONOS functionality. Therefore, SKY130 is quite a flexible technology, providing the IC engineer with a wide range of design choices. The foundry even leaves open the possibility of enabling further customization, through addition of specialized materials, such as Nb, Ge, V2O5, Carbon Nanotubes.

This marks the last brick needed for truly open hardware down at the nanometer scale.

5 Analog IC Design Course

Since the device physics is more tightly related to analog design, it makes sense to the analog design course first. The availability of the free PDK helps here, as well, even though circuit theory is the main topic of the course. The open access to all levels of the node allows presenting and sharing all the details of the fabrication, such as process stack diagram in Fig. 1 [14], as demonstrated in this paper.

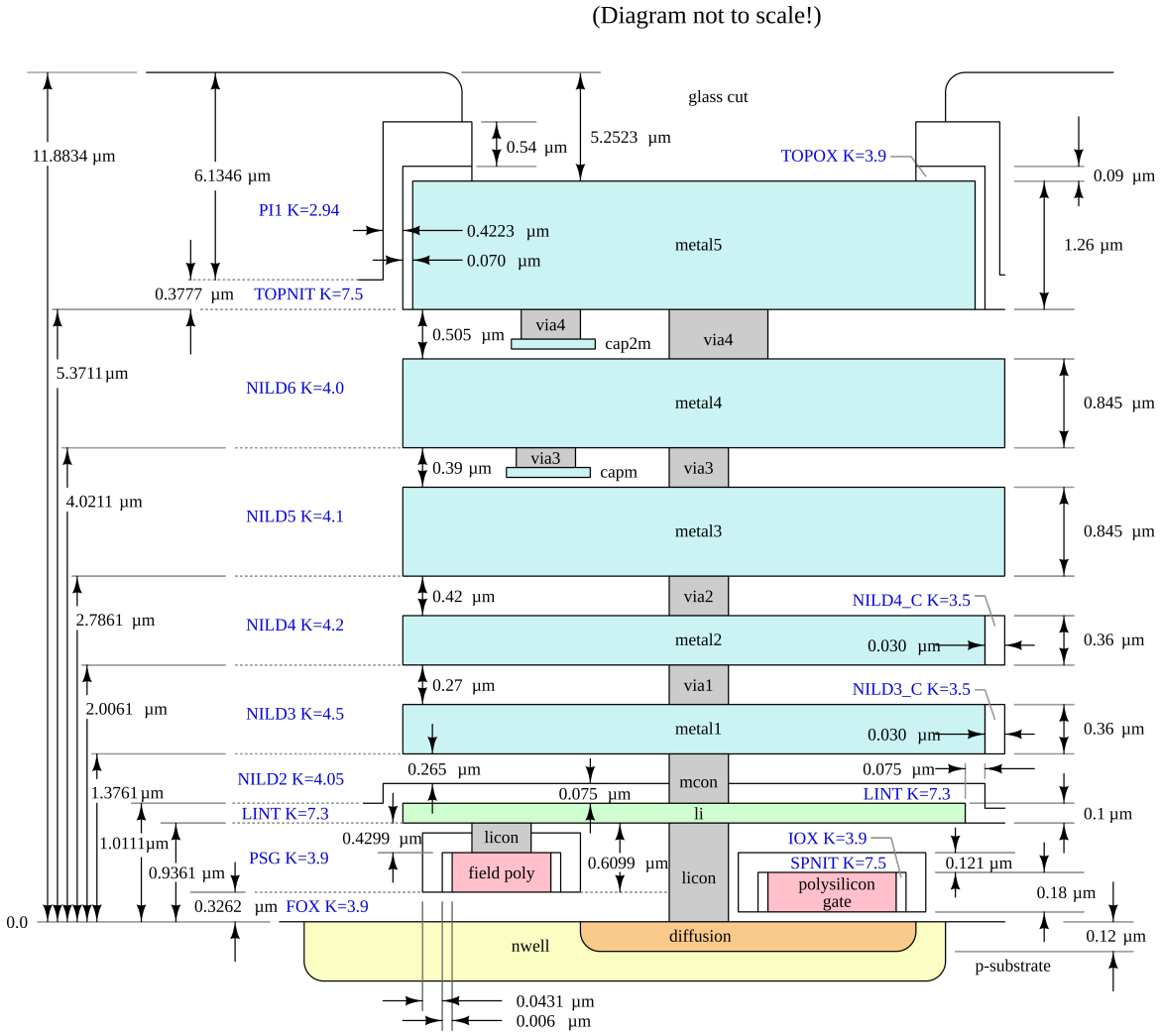

Figure 1: Process stack diagram [14]

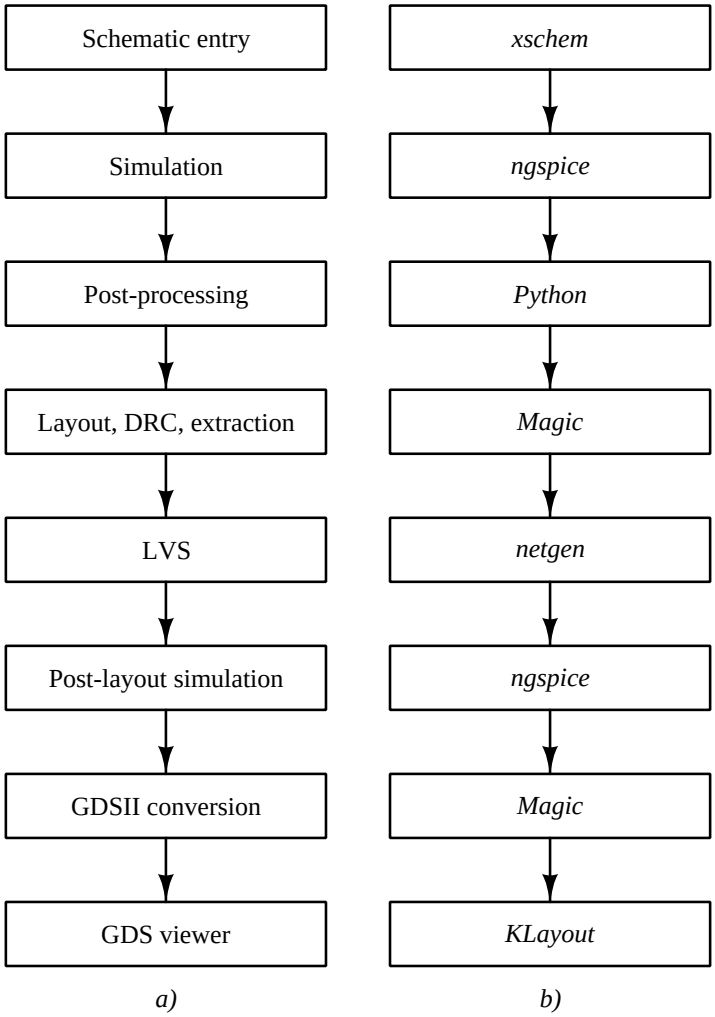

Figure 2: Analog IC design open source toolchain: a) general approach, and b) tools we applied

The toolset is different for analog and digital flows, so in this section the subset of tools we chose to use for analog circuit design is shown, Fig. 2. In the current iteration, the schematic entry is enabled by xschem [15]. For simulations, pre- and post-layout, nsgpice [9] is used. As a waveform viewer gaw [16] is fast and capable, but it is strongly suggested to have python installation handy, due to the requirements of certain simulation results processing and the agility found in countless libraries written in python. Layout is drawn in Magic [17], which provides a DRC in real time and parasitic extraction, while also converting the drawn layers into GDSII format for fabrication. LVS checks are done by netgen [18]. Finally, to make sure that GDS holds appropriate information, KLayout [19] may be used as a viewer.

Over the course of the semester, we first cover the frontend tools - beginning with the operating system; namely, at the Department of Electronics, this is the first course where students face Linux OS. Then we proceed to present basics of simulation, by working directly with the simulator, i.e. no GUI. The next stage is to discuss different device models, and learn how to manually process simple simulation results and show them graphically using python libraries. Only after the students get a grasp on the back end tools, we move to layout and LVS and talk further about DRC. Final stage of the course includes a project where a standard operational amplifier is provided as a template of very basic performance, the whole design is available in [5]. The students are tasked of improving that particular template to a given metric, e.g. to make it either more energy efficient, faster, higher magnitude, wider bandwidth, etc. There are other advantages of this course, but these are beyond the scope of this paper.

6 Digital IC Design Course

For the digital design course, we've decided to turn things the other way: we do the tools at the end, while working first on the system development first. Systems of processor core complexity may not be widely accepted as a subset of a VLSI dedicated course. Nevertheless, thanks to the vast offering of so many different classes of free IP of this level, we realized that it is actually possible to embed small steps into the computer architecture world within the digital design course. This has been demonstrated in practice, as we have been working with processor cores within this course for two years already and students' reactions are nothing but positive.

We provide thorough details on each of the cores taught in our recent publication [4], while here we only mention them: Hack [20], Sodor [21] and [22]; and point out that we have included a Chisel [23] hardware construction language in our curriculum, as well as that the latter two cores are RISC-V [24] compatible. In these efforts, resources available at the UC Berkeley courses webpages have helped greatly [25].

Once the design is settled from the computer architecture point of view, i.e. either we've reached the requirements or the time is simply up, we turn to the free tools for digital design flow That is a set of software tools used to transform the Verilog (emitted by Chisel in this case) netlist into a physical digital circuit, i.e. a set of IC layout masks in SKY130. The tools we used are the same collection used in [10]. Qflow [26] is a complete, open-source toolchain for synthesizing digital circuits starting from Verilog source and ending in physical layout for a specific target fabrication process. While the process is more detailed, we present a simplification describing only the major steps - shown in Fig. 3.

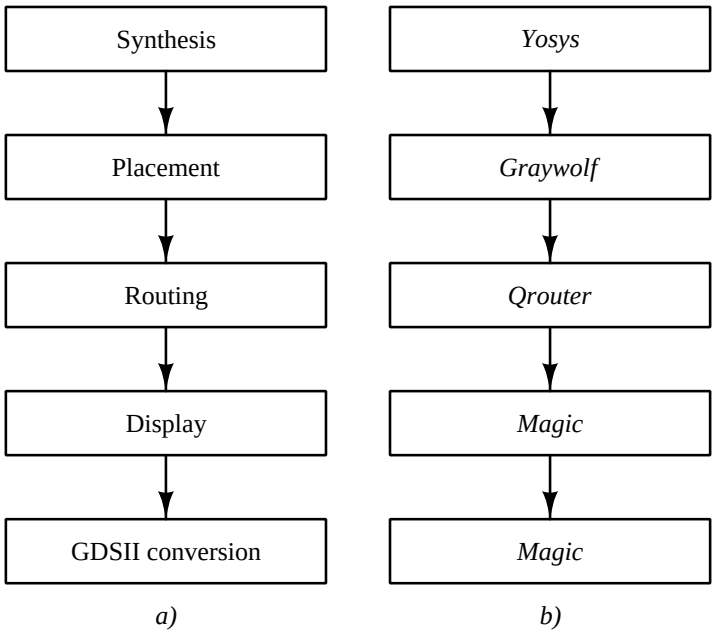

Figure 3: Digital IC design open source toolchain: a) general approach, and b) tools we applied

The first step in the automation process is to map the netlist onto a standard cells library. This step is done by yosys [27]. Next, the design is to be placed and routed. In shortest terms, this when the standard cells are spread across the available area, while grouped in blocks and interconnected (routed). Graywolf [28] is the member of the qflow toolchain that does the placement, while routing is performed by qrouter [29]. Finally, for layout inspection, DRC and GDS generation Magic [17] is used. Qflow, nor the provided PDK are not capable of creating a microprocessor that may compete with current 3 GHz+ multicore server processors, but these tools will successfully handle simpler designs that may be found in SoC all over the market - such as SPI, for example.

7 Conclusion

This is the point where all three requirement for free silicon meet: knowledge, tools and PDK, meaning that the age of open ASIC has begun. Further, both barriers have been removed, thus we may stand on the shoulders of giants to bring actual free silicon to our students. While having fabricated two analog designs using the free tools and a proprietary PDK, we hope to leverage our experiences and enable our students to fabricated completely free and open ASIC in future. In the next iteration of the courses, we hope to replace qflow with openLane and include openRAM into the curriculum.

References

[1] A. Pajkanovic, "On the application of free CAD software to electronic circuit curricula," in *Proc. of 3rd International Conference on Electrical, Electronic and Computing Engineering IcETRAN 2016*, Zlatibor, Serbia, 2016, pp. ELI1.3.1–4.

[2] A. Pajkanovic and Z. Ivanovic, "A report on recent development in application of free CAD software to IC curricula," in *Proc. of 5th IcETRAN 2018*, Palic, Serbia, 2018, pp. 847–851.

[3] A. Pajkanovic, "CMOS IC from schematic level to silicon within IC curricula using free CAD software," in *Proc. of INDEL 2020*, Banja Luka, Bosnia and Herzegovina, 2020.

[4] ——, "Free/open source EDA tools application in digital IC design curricula," in *Proc. of IcETRAN 2021*, Stanisici, Bosnia and Herzegovina, 2021.

[5] ——, "Open source CMOS general purpose operational amplifier," in *Proc. of MIEL 2021*, Nis, Serbia, 2021.

[6] Y. Tsividis, "Designing analog MOS circuits at Berkeley in the mid-70s," *IEEE SOLID-STATE CIRCUITS MAGAZINE*, vol. 6, no. 2, pp. 22–24, 2014.

[7] efabless. [Online]. Available: https://www.efabless.com

[8] Open circuit design. [Online]. Available: http://opencircuitdesign.com/

[9] Ngspice. [Online]. Available: http://ngspice.sourceforge.net

[10] T. Edwards and M. Kassem, "The Raven Chip: First-time silicon success with qflow and efabless," in *Free Silicon Conference 2019, FSiC 2019*, Paris, France, 2019. [Online]. Available: https://wiki.f-si.org/index.php/FSiC2019

[11] E. Humenberger, "ASICone. goals, timeline, participants and tools," in *Free Silicon Conference 2019, FSiC 2019*, Paris, France, 2019. [Online]. Available: https://wiki.f-si.org/index.php/FSiC2019

[12] T. Ansell and M. Saligane, "The missing pieces of open design enablement: A recent history of Google efforts," in *ICCAD '20*, Virtual Event, USA, 2020.

[13] SkyWater. [Online]. Available: https://www.skywatertechnology.com/

[14] skywater-pdk. [Online]. Available: https://github.com/google/skywater-pdk

[15] xschem. [Online]. Available: https://xschem.sourceforge.io/

[16] gaw. [Online]. Available: https://gaw.tuxfamily.org/

[17] Magic. [Online]. Available: http://opencircuitdesign.com/magic

[18] netgen. [Online]. Available: http://opencircuitdesign.com/netgen

[19] KLayout. [Online]. Available: https://www.klayout.de/

[20] nand2tetris. [Online]. Available: nand2tetris.org

[21] riscv-sodor. [Online]. Available: https://github.com/ucb-bar/riscv-sodor

[22] rocket-chip. [Online]. Available: https://github.com/chipsalliance/rocket-chip

[23] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Avižienis, J. Wawrzynek, and K. Asanović, "Chisel: Constructing hardware in a scala embedded language," in *DAC2012*, San Francisco, USA, 2012.

[24] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanović, "The risc-v instruction set manual, vols. i-ii," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-118, May 2016.

[25] Computer Architecture 152/252A, spring 2021. [Online]. Available: https://inst.eecs.berkeley.edu/~cs152/sp21/

[26] Qflow. [Online]. Available: http://opencircuitdesign.com/qflow/

[27] C. Wolf, "Yosys open synthesis suite," http://www.clifford.at/yosys/.

[28] "Graywolf," https://github.com/rubund/graywolf.

[29] "Qrouter," http://opencircuitdesign.com/qrouter.