Google/SkyWater and the Promise of the Open PDK

R. Timothy Edwards
SVP Analog and Platform
Efabless
San Jose, CA, USA
tim@efabless.com

***Abstract*—For over twenty years, silicon foundry Process Design Kits (PDKs) have been a domain of secret knowledge, non-disclosure agreements (NDAs), license servers, and password-protected download sites. The lack of transparency is at odds with today's ever-expanding universe of open source software, leading to an unusual niche for licensed commercial tools, and a very difficult space in which to explore, grow, and diversify. To overcome the problems inherent in this arrangement, the SkyWater foundry has opened up its process description to the public, a process driven and underwritten by Google and supported by Efabless and a consortium of small companies and university groups [1]. This presentation highlights the implications of having a free and publicly-accessible foundry process description on the small world of open source electronic design automation (EDA) tools for custom silicon design, how the open source PDK repository is beneficial to the worldwide chip design community, how developers of open-source EDA tools can take advantage of it, and where to move forward in a future of open-source tools and hardware.**

***Index Terms*—PDK, open source, EDA tools, foundry, VSLI**

I. Introduction

The Google/SkyWater Open Source PDK [1] was launched in July of 2020 with a repository on github under google/skywater-pdk. It was launched in conjunction with a talk by Tim Ansell to start the FOSSi foundation "Dial-up" series of presentations [2]. Other than that, it launched with relatively little fanfare due to the pandemic and corresponding lack of in-person conferences, workshops, and other events, and the difficulty of advertising an online event to a wide audience. Nevertheless, the event had several hundred participants, and the accompanying Slack and IRC channels devoted to the topic rapidly gained thousands of members [3].

The launch was preceeded by nearly a year of negotiations. Behind the idea from the beginning was the SkyWater foundry in Minnesota, one of a small and increasingly rare set of silicon foundries based in the United States. Foundries that are on the cutting edge of process technology tend to be very closed and tight-lipped and are not expected to take seriously the idea of making their foundry data public. However, those foundries that support the substantial amount of manufacturing in well established nodes (especially in the range of 0.5um down to 0.18um) have tended to be more open and user-friendly, alhough never before to the extent of releasing the technical details of a foundry process to the public. For example, X-Fab, which provides processes in the 0.35um to 0.18um nodes, has been generous in allowing Efabless to make a number of its IP libraries user-facing on the cloud-based efabless.com platform, although some technical details remain hidden. The open SkyWater process is a 130nm node, well suited to both analog and (modestly sized) digital designs, and is a feature size that has been around since 2001. Process options include deep n-well, MiM capacitors, high and ultra-high sheet ρ resistors, and isolated gate flash memory transistors (Fig. 1).

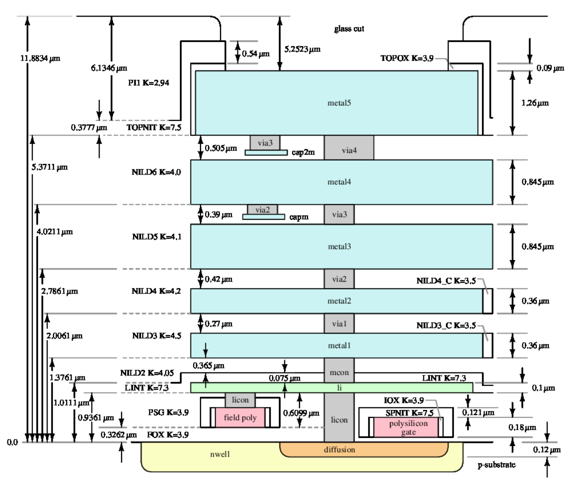

Fig. 1. Process stack of the SkyWater 130nm open process.

While the competitive nature of cutting-edge processes means that the newest and most expensive processes will remain proprietary and very closely guarded, there is a clear business model recognizing the advantages of making older and well established processes public knowledge. There is little competitive advantage to be gained from keeping secret information that is generally constrained by the machinery used and therefore easy enough to duplicate. Certainly each PDK represents a large amount of work by the foundry in refining design rules and characterizing and modeling devices. At some point, though, the problem becomes one of maintaining non-disclosure agreements, tracking customers, providing a large amount of customer support, and maintaing the PDK. Dowload sites must be provided and maintained, errors must be fixed, and documentation must be written and updated. The foundry is generally considered responsible for providing useful IP to the end user, including digital standard cell libraires, padframe I/O libraries, RAM and ROM compilers, and critical analog circuits such as crystal oscillators, voltage regulators, bandgap references, and power-on-reset generators. Much of this work has very little to do with day-to-day foundry operations of manufacturing, and represent a huge overhead.

II. Why it Matters

The idea behind open-sourcing a PDK is to offload a large number of tasks related to EDA tool support. By making the repository publicly accessible, a community of end users can provide mutual support for understanding how to use the process with numerous tools. It is no longer the responsibility of the foundry to provide this user support, maintain the support for tools, or manage user submissions for multi-project wafer runs. Errors that are found by the community will be fixed by the community. Missing functionality will be provided by the community as well. In the spirit of open source, the vast majority of this support and IP will itself be provided under open source licensing.

For the designer, the gains are very clear: There is no longer a need to sign NDAs, and no concern over what information may or may not be shared with others, especially tool developers who may need to see a failing example in order to properly debug a problem with the software. Entire designs may be published including schematics and layout without restrictions; and designs may be placed into public repositories like github and gitlab, ready to be downloaded and sent to manufacture, or used within another design, or modified, improved, and re-published as a "remix".

In this way, the design of hardware can look much more like the design of software or the design of 3D-printed things: Public, dynamic, ever changing, and ever improving. The greatest promise is that of democratization: The shifting of custom hardware design from a handful of large and established companies and well-funded research universities to individuals, small companies and startups, lesser-known universities, colleges, and even high schools. In-house design will give way to vast project collaborations spanning the globe.

III. A History of Open Source PDKs

The proprietary and closed nature of foundry processes was certainly aided and abetted by the commercial EDA tool vendors. There are only a few of these vendors, and they can maintain their dominance because support for their tools and formats is the burden on the foundry, so it is in the foundry's best interest to limit the number of tools to support.

In the early boom days of silicon manufacture, mainly through the late 1980s and early 1990s, many design tools were in fact open source, and many (SPICE, magic, VIS/SIS, etc.) were distributed freely before this became a common practice in software (all the above-mentioned tools predate Linux by at least seven or eight years). The MOSIS foundation in particular supported multiple foundries and processes with a partly open-source PDK which it called SCMOS (scalable CMOS). However, at the time, the foundry nodes represented by SCMOS were still considered advanced, and the information still tightly guarded. The SCMOS PDKs only worked because they used conservative rules that did not reveal manufacturing limits; device models and parameters were not freely available and could only be obtained through an NDA with MOSIS. While IP libraries designed with the SCMOS PDKs are still available, MOSIS has dropped support for these PDKs, making them equivalent to the so-called "academic" PDKs such as FreePDK45 which are available for use but unconnected to any specific foundry and therefore not manufacturable [4]. The Google/SkyWater PDK goes much further than the SCMOS PDKs from MOSIS ever did: Not only are the exact design rules of the foundry available, but even specialty rules such as rules for compact SRAM core cells and for flash memory cells are made public, as well. All device models and parameters, characterized at corners, are part of the repository.

In a sense, the huge amount of available data in the open source PDK provides a new challenge: The older tools that once worked with the SCMOS PDKs never had the depth and breadth of access to IP and formats. The new PDK provides everything needed for full toolchains, whether analog, digital, or mixed-signal. The older tools do not have the complete integration that the commercial EDA tools enjoy, having traditionally had only a fraction of the files and formats available to use. Much of the work on preparing and uploading the Google/SkyWater PDK into the github repository has been to prepare a complete set of files in standard, mostly human-readable, non-proprietary formats, and to organize them in ways that can be standardized and used by open source tools looking to the future. It is now needed for the open source EDA tool developers to take advantage of the organization of files and formats, and to create the tool integration into complete flows that is so badly needed.

IV. Relevance to Existing Tools

It is not strictly necessary for an open source PDK to be used with open source tools. But it is the goal of Google to make sure that open source tools are given top priority for support by ensuring that the PDK data and IP libraries are all in common formats that are well described and easy enough for an open source tool developer to code or obtain a parser for. This prioritization of common formats over proprietary formats creates an opportunity for open source EDA tool developers to fill the gaps in tool capabilities with an ease and speed not seen since the early days of EDA software.

The most obvious benefit to having an open PDK for open source EDA tool development is simply the fact that when bugs are found in the tools, an example can be sent to the developer to reproduce the error exactly, avoiding the need to obfuscate the example or find a similar example using an SCMOS or "academic" process. The lack of transparency inherent in proprietary PDKs slows open source tool development, severely restricting the gains that are expected from open source development: Rapid response with the ability to upgrade a tool or PDK immediately upon a fix becoming available. Case in point: Upon release of the "Openlane" digital flow for the SkyWater process, users discovered a problem with the "netgen" open source LVS tool, in which a symmetry-breaking routine at the end had a O(N^2) or greater execution time, causing LVS to take hours to complete. Once this issue was raised, I was able to pin down the problem, and by the next day the run-time for netgen had been reduced to a few minutes. Upon announcing the fix on the public Slack channel, the users patched their tools, and by the end of the day multiple users had confirmed the fix and moved forward with their designs, a turn-around time unheard of in the closed-source commercial world.

The second most obvious benefit of the open PDK is the lack of license servers and the overhead in maintaining them. Improperly working license servers (or simply an expired license) are an impediment to design, and can cost designers days of delay while issues are sorted out.

A third benefit to the open PDK is the ability to reach a commmuity consensus on what file formats are the best to use. This avoids the need to rely on proprietary formats, which are tightly controlled by commercial EDA tool consortiums and very slow to correct problems and adapt to new or different design methods.

The promise of open source EDA tools is already apparent in the projects associated with the roll-out of the Google/SkyWater PDK: The OpenRAM project [5] seeks to put the generation of SRAM, ROM, and flash memory into the hands of the designer, and with full visibility of the process, keep this set of critical IP blocks from being some mysterious thing to be trusted, but not inspected. The Oklahoma State University (OSU) standard cell library [6] is not only a valuable IP library, but is presented along with the complete set of tools and methodology used for design and characterization. The Openlane flow from American University in Cairo (AUC) [7] is a digital synthesis flow based around the OpenROAD tools from UCSD (and others) [8] with specific attention to established process nodes, with specific setups for the SkyWater 130nm process. Openlane has already filled in one of the outstanding gaps in open source tools by integrating design-for-test (DFT) methodology into the synthesis flow [9], and provides the ability to do chip top level assembly and signal routing. This is just a sampling of the groups starting to form a valuable ecosystem around the open source PDK.

As a demonstration that existing open source EDA tools are already capable of handling the end-to-end design of a complete custom chip, Efabless designed and produced a series of RISC-V microprocessors, collectively named "striVe," and made entirely with open source EDA tools and containing exclusively open source IP (Fig. 2). When the full set of IP libraries used by the striVe chip designs has been put in the Google/SkyWater repository, these designs will be placed in the Efabless github repository and available for anyone to download, inspect, and use.

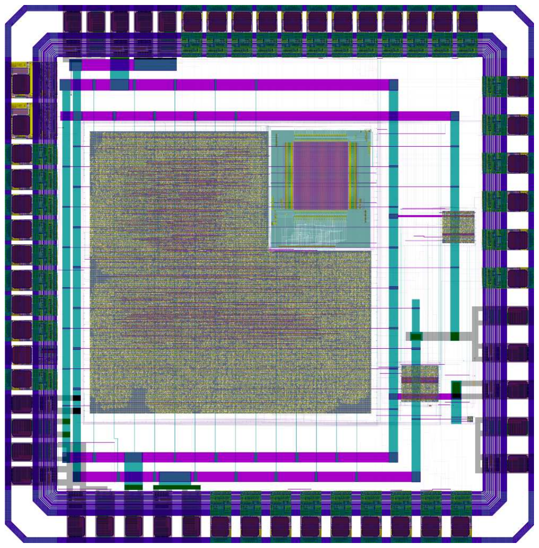
Fig. 2. StriVe2, a RISC-V SoC created with Openlane and featuring an SRAM memory block compiled by OpenRAM. Designed by the Efabless team using all open source tools, and manufactured by SkyWater foundry in the 130nm open process.

V. Relevance to Future Tools

The installation and maintenance of PDKs on the user end is an often overlooked overhead cost for chip design, but tracking updated from the foundry side and maintaining proper versioning is a large task, and usually companies or universities with site-wide licenses for running the commercial EDA tools will have a dedicated person or team devoted solely to PDK maintenance. One of the biggest challenges needed to make the open PDK ecosystem work well is an integrated system of tools for multiple work flows [11]. The Openlane digital synthesis flow cited above works well for digital synthesis and chip assembly. However, a complete ecosystem needs analog and mixed-signal flows as well, and an overall common infrastructure for the files needed both for tool setups and IP libraries. My own contribution to this is the "open_pdks" tool [10], which is a GNU automake/autoconf system utilizing various shell and python scripts to prepare the right files and environments for different open source tools such as ngspice, magic, klayout, netgen, and openlane.

While digital synthesis flows are powerful and go hand in hand with open source desigs in hardware description languages, there will always be a need for traditional analog design. The need for open source analog design flows reveals a glaring gap in the ecosystem: No common agreed-upon format for schematic editing and capture. I have been working with interns over the summer to help determine what such a format should look like, and to generate useful tools in support of analog design flows. I am leaning toward using the xschem [12] format as the "standard" schematic and symbol format, and supporting additional formats with translation scripts (yet to be written). One important project in progress defines a common set of symbols that can be prepared as a library for any of a number of schematic entry tools (xschem, xcircuit, XIC, KiCAD) and a tool to map standard technology files (liberty, LEF, SPICE) to the symbol library (Fig. 3). One intern started a project to automatically generate schematics from SPICE netlists. But a large amount of work remains to be done to define proper analog and mixed-signal design flows.

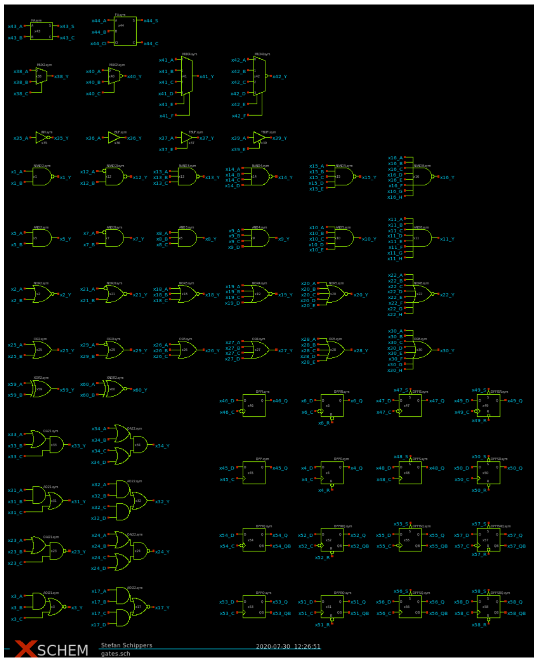

Fig. 3. A common set of symbols to map a PDK digital library to open source tools, courtesy of Stefan Schippers.

Another area where proprietary formats dominate and there are no suitable open source equivalents is design rule checking (DRC), also an important part of analog and mixed-signal design flows. The industry-standard format, Calibre, is tied to the tool that uses it, and is not even necessarily appropriate for other tools like Magic that have a fundamentally different database and use different methods for design rule checking. Also, the format is essentially a list of database operations to perform, not a description of rules, and as such is not very human-readable. But DRC rule types, especially for established nodes, are limited in number. They can and should be presented in a universal format that can be used both for generating the setup files for the tools that do the checks, and for automatically generating the documentation that describes the rules.

In both these cases (schematic files and DRC rules), an open format is best for being adaptable to the needs of different tools and different users. It is important not to "lock in" a format but let it evolve with the needs of the design community.

The Google/SkyWater open-source PDK is not merely an academic exercise. Google is underwriting multi-project wafer (MPW) runs on the SkyWater process with the express purpose of encouraging designers to use the process and the open PDK to explore the boundaries of the design space. Management of the design submissions falls to Efabless, where we have prepared a one-size-fits-all chip "harness," or container (called "Caravel", after the type of sailing ship used for early exploration and trade) for holding user designs. The MPW run management itself provides new challenges for open-source EDA tools, since automatic fill pattern generation and optical correction are topics we may need to consider, but have not been in the scope of open source tool development. The sheer amount of data involved in the preparation of an MPW is also likely to stress the limits of a number of tools.

VI. Conclusions

The Google/SkyWater 130nm open PDK is now available to the public on github as the first ever fully open-source silicon foundry process description, including fully open-source IP libraries in formats suitable for open-source EDA tool design flows. A thriving ecosystem of tools and tool development is already forming around this online offering, aided by the promise of unrestricted access and a new ability to experiment and explore the possibilities and limits of a foundry process without the burden of NDAs, closed-source software, and limited available IP from "trusted" vendors. The current state of EDA tool development for all types of design flows is rapidly progressing, led by digital synthesis tools. A large amount of work remains to be done for integrating tools, methods, and formats for analog and mixed-signal design flows, and for full-chip assembly and design verification. The community must ensure that these tools and formats remain fluid and adaptable, such that the promise of open-source hardware can match the universally-understood benefits of open-source software.

Acknowledgment

I would like to thank Jeff Carr of Wit for getting the ball rolling, Mohamed Kassem of Efabless for keeping it moving, and to Tim Ansell of Google for seeing the project through to fruition. But a hundred voices made it happen, and there is not enough room to thank them all.

References

[1] Tim Ansell, "Google/Skywater open PDK," https://github.com/google/skywater-pdk
[2] The FOSSi Foundation, "FOSSi Dial-Up," https://fossi-foundation.org/dial-up/
[3] "Google + SkyWater FOSS 130nm Production PDK," https://join.skywater.tools/
[4] "N.C. State University FreePDK45," https://research.ece.ncsu.edu/eda/freepdk/freepdk45/
[5] M. R. Guthaus, J. E. Stine, S. Ataei, B. Chen, and B. Wu, M. Sarwar, "OpenRAM: An Open-Source Memory Compiler," Proceedings of the 35th International Conference on Computer-Aided Design (ICCAD), 2016.
[6] James Stine, "Oklahoma State University System on Chip (SoC) Design Flows," https://vlsiarch.ecen.okstate.edu/flow/
[7] Mohamed Shalan, "OpenLANE," https://github.com/efabless/openlane
[8] T. Ajayi, V. A. Chhabria, M. Fogaa, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem, G. Pradipta, S. Reda, M. Saligane, S. S. Sapatnekar, C. Sechen, M. Shalan, W. Swartz, L. Wang, Z. Wang, M. Woo and B. Xu, "Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project," Proceedings of the ACM/IEEE Design Automation Conference, 2019.
[9] Mohamed Gaber, Manar Abdelatty and Mohamed Shalan, "Fault, an Open Source DFT Toolchain," Proceedings of the Workshop on Open-Source EDA Technology (WOSET19), November 2019.
[10] R. Timothy Edwards, "Open_PDKs PDK Installer for open-source tools," http://www.opencircuitdesign.com/open_pdks
[11] R. Timothy Edwards, "The New Golden Age of Open Silicon," Keynote presentation, Workshop on Open Source EDA Technology (WOSET) 2019.
[12] Stefan Schippers, "XSCHEM: schematic capture and netlisting EDA tool," https://xschem.sourceforge.io/stefan/index.html