How to Make Open-Source EDA Project Sustainable: a Perspective from Industry

Frank Liu

IBM Research, Austin, Texas
Department of ECE, Texas A&M University, College Station, Texas

ABSTRACT

With the wide availability of software collaboration platforms such as Github and Bitbucket, the threshold of starting a new open source project is low. However to ensure an open source project success and having long-lasting impact, it is crucial to maintain its sustainability. In this white paper I will outline some observations on the sustainability of open source EDA projects.

1. BACKGROUND

Open source has proven to be a viable software development model for decades. With the introduction of RISC-V[1], ISA is posed to move into open source development model. Given the indispensable role of EDA of linking the hardware design and the final VLSI product, it's a natural question when EDA software and environment will adopt the open source model soon. Interestingly one of the earliest EDA software packages, SPICE, can be considered as an open source project[2]. Because the source code of SPICE is available to the public, it can be freely modified to accommodate various deployment platforms[3] and to have enhanced functionalities[4, 5]. More importantly, commercial companies are free to improve and package SPICE into their proprietary commercial products. To some extent, the openness of SPICE is one of the major reasons of its success and long lasting impact to the EDA and semiconductor industry. Although the events occurred before open source movement is formally established, the success of SPICE clearly demonstrated the benefits of software transparency.

In the past few decades, the software industry has embraced the open source model. Nowadays, open source has penetrated almost all aspects of software, from operating system (Linux), big data platforms (Apache Hadoop, Spark[6]), scientific computing(R[7], Octave[8]), to sophisticated application software (openFOAM[9]) and deep learning frameworks(Caffe, Tensor Flow[10]). The code transparency enforced by the open source model can improve security, but more importantly it can drastically increase interoperability and encourage collaboration. Nowadays it is common to see multiple companies pool the resources together to develop a common infrastructure code. For example, many software companies collaborate on developing Linux kernel[11], which is a drastic departure from the early days when major software companies developing their own own proprietary Unix-like operating system. By sharing the development cost, each company can concentrate on the customization of the OS on their own hardware platform. The transparency also enables rapid adaptation. One example is the development of Android OS for mobile devices, which is based on Linux. For academic researchers, the open source projects provide channels to prototype and demonstrate the latest research outcome on real-world applications, instead of small scale toy examples.

However, although there have been many discussions among EDA researchers and practitioners on open source EDA, the open source model has yet to become mainstream in EDA. By piggybacking on the success of open source movement, nowadays it is easy to start an open source project on widely available software collaboration platforms such as github. The key to make an open source project successful is to ensure its *sustainability*, in the sense that there is sustained enthusiasm and resources to maintain its viability and growth. In the remaining part of this white paper, I will discuss issues on community, business model as well as some technical issues.

2. COMMUNITY

To make an open source project sustainable, there must exist a community of developers who can and will actively participate in the project. High quality software requires dedicated effort to design, implement, test and support. Without an active community, it would be difficult to even maintain a software project, let alone further develop it. For example, between 2005 and 2013, over 9,000 software developers have contributed to the Linux kernel. Although approximately one third contributed only one patch, top 30 developers contributed 18% of the total, while the top 10 developers contributed 8.4%[11]. It is safe to claim that without those key contirbutors, it would be very difficult to sustain the project. Although some of the contributors are hobbists, many active Linux contributors are professional software developers from tech companies such as Red Hat, Intel, Google and IBM. On average, over 1,600 developers contributed to each Linux kernel version[12]. Besides the traditional software development and maintanance, it is crucial for each community to have a core team, who have a broad understanding of the invidivual project, and are capable of collectively making long term decisions on the future directions of the invividual project.

A challenge for open source EDA projects is to develop and maintain such an active community. In the Linux development community, the lion share of the contributors are professional software developers. On one hand, many core EDA algorithms in EDA software are highly specialized (not commonly covered by traditional undergraduate-level engineering cirricula). On the other hand, if we use the number of graduate students in EDA as a metric to measure the student involvement, the supply side of fresh CS students who are proficient in EDA is not as strong as in other fields such as Big Data and machine learning. Moreover, many EDA applications place a premium on performance (latency, throughput and memory footprint), which means EDA software requires more tuning and requires a dedicated core team. The compounding effect is that the number of qualified contributors is limited. The bright spot is that as open source EDA projects take root, there is the potential of attracting more fresh students.

3. BUSINESS MODEL

As alluded earlier, many open source Linux contributors are in fact professional software developers. Hence their employers are effectively supporting the development of Linux kernel. Among general public, there is a misconception that since open source software is freely available to individual consumers, hence there is no cost in developing it. Reality cannot be more different. Any good software requires time and effort to develop and maintain, the same is true for open source software projects. According to an early study[13], the total cost of developing and supporting a Linux distribution is estimated to be around \$1.2B. Without a sustainable business model to generate a continuous stream of funding, an open source project can quickly wither. Moreover, a sustainable business model is also crucial to maintain a coherent core of community developers. There are a few business models used by other successful open source projects.

One example is Linux, where many companies see the advantage of collaborating on a common operating system, and are willing to pool their professional developers into this project. The contributing companies range from device manufacturers, microprocessor vendors, system vendors, to IT service companies and cloud providers. This is manifested by the wide spectrum of the member companies of Linux Foundation. Even among the companies who are competing against each other, there is still the benefit of "competitive collaboration": by contributing to Linux, a company can concentrate its resources on differentiable technical advantages, instead of duplicating effort to develop common features. Another reason is that Linux has become so dominant that no company can ignore it. Besides Linux, many Big Data application software from Apache Foundation follow this model as well.

A different model is Genome Analysis Toolkit (GATK), which is almost entirely developed by Broad Institute, a nonprofit research organization. Although some commercial companies such as Intel, Google, Cloudera and IBM contribute, the overall software architecture and many core algorithms are developed by Broad. Similar to many EDA software, genomics analysis is also fairly complex and specialized. By having a big team of genomics software developers, Broad can quickly incorporate the best and latest analysis methods into a product for its big army of biology researchers to use. Since it is a nonprofit organization, it also opens the software for every biology researcher and genomics clinician to use, free of charge. The majority of the development cost is hence from philanthropic donations to Broad Institute. Over the years, GATK has become the de-facto industry standard for a genomics analysis. The sequencing equipment manufacturer and clinical genomics hospitals are willing to contribute to maintain its development. Interesting the situation of GATK is somehow similar to SPICE.

Another business model is to provide customization, service and support. Enterprise customers expect certain level of support, in the format of customization to their platforms, bug fixes, and assistance in integrating the open source solutions into their own platforms. Once a particular open source solution is adopted by an enterprise customer, there is a certain product time window in which the solution has to be supported. An open source project can sustain itself by provide such paid service and support.

For open source EDA projects, the exact sustainable business models are yet to be defined. At the end of the day, a software project (whether open sourced or proprietary) can only survive if it's useful to the end customers. Commercial EDA software vendors have been working with customers for decades and have good understanding on the needs of the end customers. It would be desirable if the open source EDA community can find a collaboration model with the commercial vendors.

4. AGILE DEVELOPMENT AND COMMON BENCHMARKS

Many open source projects leverage contemporary agile software development. For example, Travis CI is closely integrated in github[14]. The deployment of agile model not only improves the communication among the contributors (who are usually geographically dispersed), but also improves the overall software quality. There are a plethora of available materials on how to deploy agile model.

For highly specialized application EDA software, the important of a common dataset for benchmarking cannot be overstated. There are well-known ISCAS benchmarks[15], as well as the GSRC Bookshelf[16]. To make the open source EDA projects have more impact, it would be desirable to ensure the benchmark datasets close to real applications.

5. PDK INTERFACE

The contemporary VLSI design flows, particularly the backend, have strong dependency on the semiconductor manufacturing process. Ideally the PDK (Process Design Kit) should be open to realize a complex design at a state-of-the-art technology note, which is the best way to fully demonstrate the true value of an open source EDA project. Although there are some academic efforts in this direction[17, 18], however most foundry PDK's are well guarded under NDA. The reason is fully understandable: a PDK contains sufficient trade secrets that making it public is unwise. A possible compromise is to establish an intermediate layer which can provide sufficient information to the design tools, while masquerading technology secretes.

6. INTEROPERABILITY

A contemporary EDA flow is highly complex. To finish a design, many unique components have to be used, often in an iterative fashion. Interoperability of these components is crucial to ensure the performance. An interesting observation is the *netlist* format defined by SPICE. Although it is not particularly efficient, it is human readable and has been used extensively for various EDA software packages. A parallel scenario is the data formats used in genomics. The original FASTQ[19] and SAM[20] format are based on plain text and human readable. As genomics research and application advanced, the data volume became a big issue. Hence more advanced features (e.g., compression and encryption) were introduced to mitigate these concerns. Although this approach seems to be quite inefficient, however it can be an interesting way to ensure the openness of the data format.

7. CONCLUSION

Open source has proven to be a viable model for software development. Sustainability is crucial for an open source project to be successful and have long-lasting impact. Issues such as cultivating an active community and identifying a viable business model are crucial to maintain the sustainability of open source EDA model, which is essential for its success.

8. REFERENCES

[1] "RISC-V Specifications." `https://riscv.org/specifications/`.

[2] "README Spice3f4." `https://ptolemy.berkeley.edu/projects/embedded/pubs/downloads/spice/spice3f4.readme`, 1993.

[3] J. Puhan, T. Tuma, and I. Fajfar, "Spice for windows 95/98/nt," The University of Ljubljana, Slovenia, at the Faculty of Electrical Engineering, 1998.

[4] K. Tseng, C. Foo, and P. Palmer, "Implementing power diode models in SPICE and Saber," in Power Electronics Specialists Conference, PESC'94 Record., 25th Annual IEEE, vol. 1, pp. 59–63, IEEE, 1994.

[5] F. Fiori, T. Foti, and V. Pozzolo, "Computer aided analysis of RF effects in BJT circuits," in Electromagnetic Compatibility, 1996. Symposium Record. IEEE 1996 International Symposium on, pp. 294–299, IEEE, 1996.

[6] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets.,"

[7] R. C. Team et al., "R: A language and environment for statistical computing," 2013.

[8] J. W. Eaton, D. Bateman, and S. Hauberg, Gnu octave. Network thoery London, 1997.

[9] H. Jasak, A. Jemcov, Z. Tukovic, et al., "Openfoam: A c++ library for complex physics simulations," in International workshop on coupled methods in numerical dynamics, vol. 1000, pp. 1–20, IUC Dubrovnik, Croatia, 2007.

[10] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "Tensorflow: a system for large-scale machine learning.," in OSDI, vol. 16, pp. 265–283, 2016.

[11] S. Shankland, "Linux development by the numbers: Big and getting bigger." `https://www.cnet.com/news/linux-development-by-the-numbers-big-and-getting-bigger`, 2013.

[12] "The Linux Foundation." `https://www.linuxfoundation.org/projects/linux/`, 2018.

[13] A. McPherson, B. Proffitt, and R. Hale-Evans, "Estimating the total cost of a Linux distribution." `https://www.linux.com/publications/estimating-total-cost-linux-distribution`, 2008.

[14] B. Vasilescu, Y. Yu, H. Wang, P. Devanbu, and V. Filkov, "Quality and productivity outcomes relating to continuous integration in Github," in Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, pp. 805–816, ACM, 2015.

[15] D. Bryan, "The ISCAS'85 benchmark circuits and netlist format," North Carolina State University, vol. 25, 1985.

[16] A. E. Caldwell and I. L. Markov, "Toward CAD-IP reuse: A web bookshelf of fundamental algorithms," IEEE design & test of computers, no. 3, pp. 72–81, 2002.

[17] "FreePDK." `https://www.eda.ncsu.edu/wiki/FreePDK`.

[18] M. R. Guthaus, J. E. Stine, S. Ataei, B. Chen, B. Wu, and M. Sarwar, "OpenRAM: An open-source memory compiler," in 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–6, Nov 2016.

[19] P. J. Cock, C. J. Fields, N. Goto, M. L. Heuer, and P. M. Rice, "The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants," Nucleic acids research, vol. 38, no. 6, pp. 1767–1771, 2009.

[20] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, and R. Durbin, "The sequence alignment/map format and SAMtools," Bioinformatics, vol. 25, no. 16, pp. 2078–2079, 2009.