Developing an Open-Source Silicon Ecosystem: The Silicon Commons

Andreas Kurth[1], Marno van der Maas[1], Gregory Chadwick[1], lowRISC contributors*

[1]lowRISC CIC, Cambridge, United Kingdom

Abstract

Open-source silicon has the potential to reduce the blind-trust risk in integrated circuits (ICs) and enable reuse and collaboration on high-quality, thoroughly verified, interoperable IC blocks. RISC-V as an open standard instruction set architecture (ISA) is a key enabler for open-source silicon. The RISC-V ISA and an open-source RISC-V core alone are not sufficient to form an open-source silicon ecosystem, however: other IC blocks, such as on-chip interconnects and off-chip interfaces adhering to industry standards, are also required. While there is open-source design code for some of these blocks, practically none of them come with verification code, which is crucial for commercial silicon. The Silicon Commons is an open-source silicon ecosystem that contains design and verification code of IC blocks that form a full chip framework. The Silicon Commons also standardizes technical and organizational processes that enable multiple organizations to collaboratively engineer silicon and software that serves their mutual interests. The Silicon Commons is actively applied in the OpenTitan project.

Introduction

Open-source silicon—which we define as permissively licensed logical IC design that anyone can inspect, modify, or contribute to—has great potential: Firstly, it increases transparency and thus reduces blind-trust risks, allowing independent and/or public review and certification of an IC. Secondly, it promotes high-quality design because stakeholders share an interest in adherence to standards and best practices and because the open-source code is subject to public scrutiny. Thirdly, it creates economic flexibility and opportunities because diverse organizations can collaborate in an ecosystem that does not solely rely on the support of an individual organization.

The Silicon Commons is an open-source silicon ecosystem. It not only contains a set of existing design and verification code of IC blocks that form a full chip framework but also standardizes technical and organizational processes to *maintain and evolve* that code, which are crucial to make the ecosystem *sustainable*. This is a key difference between the Silicon Commons and the practice of releasing pre-developed silicon code—sometimes even only design and no verification code—under an open-source license.

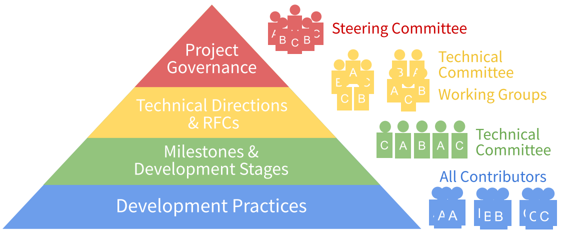

Figure 1: *The four layers of the Silicon Commons.*

The four layers of the Silicon Commons are shown in Fig. 1. Each layer involves persons from multiple organizations, as indicated by the letters. Development practices, ranging from the contribution workflow to a common design verification framework, form the basis. Milestones and development stages ensure progress towards project goals (e.g., a tapeout) as well as project-wide quality control through sign-off checklists. Technical directions and specifications as request for comments (RFCs) establish consensus on important technical decisions. Project governance is responsible for managing budget and funding as well as defining the project organization and the technical roadmap.

This article focuses on selected development practices of the Silicon Commons.

Code Base and Contribution Workflow

A public repository on which active development happens is central for a Silicon Commons project. Not only code but also issue tracking and code reviews have to be public, so that assessments, decisions, and known issues are publicly visible, too. These are crucial for external parties to build trust in the quality and progress of the project. In a silicon project, there may be code related to proprietary technology that cannot be open-sourced. Such code can be isolated by creating open-source *wrappers* that implement the necessary behavioral subset for simulation.

The contribution workflow of the Silicon Commons, shown in Fig. 2, ensures quality and consensus. It often starts with the creation of an issue to specify a problem together with a proposed solution. This makes the issue visible throughout the project and starts a discussion to find consensus. If the issue is

*Contact email: `info@lowrisc.org`

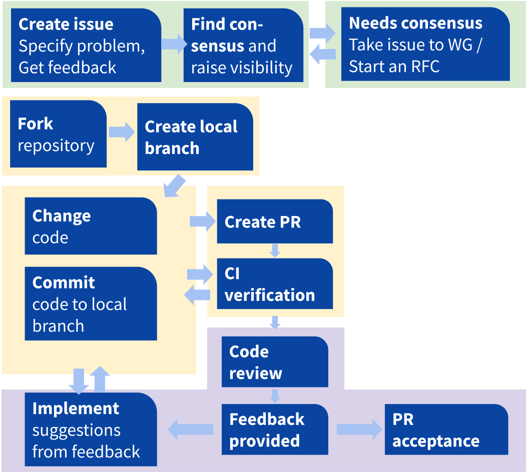
Create issue
Specify problem, Get feedback
Find consensus and raise visibility
Needs consensus
Take issue to WG / Start an RFC
Fork repository
Create local branch
Change code
Commit code to local branch
Create PR
CI verification
Code review
Implement suggestions from feedback
Feedback provided
PR acceptance

Figure 2: *Contribution workflow.*

a complex or controversial one, it may have to be resolved by a workgroup or through an RFC. When a solution is agreed upon, an engineer (who does not have to be a member of a partner organization) implements it and creates a pull request (PR). This PR is subject to automated checks and human code review. When a PR passes all checks and got approved by a *committer*[1], it can be merged to the shared main branch.

Code reviews are a central benefit of open-source silicon: *every* contribution goes through code review. They serve multiple purposes: ensure the quality of the shared code base, ensure consensus around changes, and spread knowledge and awareness about changes. The goal of each code review is to ensure that a contribution improves the overall quality of the shared code base (and *not* to ensure that a contribution is subjectively perfect). A code review should cover aspects such as design, functionality, complexity, tests, documentation, and coding style.

Coding style[2] is often overlooked for silicon code, even though it is very important to ensure quality of results and compatibility with various EDA tools. For instance, FSMs should be written in a specific way to get correctly handled by synthesis as well as coverage analysis tools. Homogenizing the format of source code ensures quality by enforcing best practices and allows engineers to focus on semantics and understand code more quickly and thoroughly.

Comportable IC blocks adhere to certain requirements so that they “plug and play” with other comportable IC blocks to form a full chip framework. These requirements include a machine-readable specification file that defines clocking, resets, on-chip interfaces, addressable registers, and optionally interrupts, alerts, and non-standard inter-module signals. From this specification, Silicon Commons tools generate hardware code to implement the registers and verify parts of the block as well as software code so that a processor can control the block.

For the hardware/software interface, the Silicon Commons defines *device interface functions (DIFs)*, which make it easy to use hardware from software for its intended purposes. DIFs can be seen as a working code reference for interacting with an IC block but they should not be considered device drivers: they target design verification as well as pre- and post-silicon validation rather than end-user safety.

Continuous Integration and Verification

Continuous integration (CI) reduces the risk of divergence and integration conflicts by developing self-contained, small changes and merging them as soon as possible. CI is usually more difficult for silicon projects than for software projects, because a lot of code has to be written before a non-trivial IC block has all its basic functionality. The Silicon Commons helps implement CI for silicon projects mainly in three ways: Firstly, its processes help break the work for a complex change into work items that can be completed in a day and reviewed in less than an hour. Secondly, IC blocks that are comportable benefit from generated code for interfacing and verifying the block. Thirdly, it provides tooling for automated quality assurance (from code linting to functional verification on FPGA), automated feedback on PRs, and automated building and provisioning of artifacts such as FPGA bitstreams.

Summary

The Silicon Commons is an open-source silicon ecosystem that contains IC blocks that form a full chip framework—including the specification, design code, and verification framework for individual blocks and entire chips—as well as the technical and organizational processes that enable multiple organizations to collaboratively engineer silicon projects to their mutual benefit and that sustain the ecosystem. The Silicon Commons has been developed and is actively applied in OpenTitan[3], an open-source project building transparent, high-quality silicon root of trust chips.

Author Biographies

A. Kurth is Senior Engineer at lowRISC and holds a Doctor of Science from ETH Zurich.
M. van der Maas is Senior Engineer at lowRISC and holds a PhD from Cambridge University.
G. Chadwick is Digital Design Lead at lowRISC and holds a PhD from Cambridge University.

[1] *Committers* are experienced contributors to whom the Technical Committee has granted write access to the code base. They can approve and merge contributions from others but need another committer to approve their own contributions.

[2] See `https://github.com/lowRISC/style-guides` for our silicon design and verification code style guides.

[3] `https://opentitan.org`