Wiring Circuits Is Easy as $\{0, 1, \omega\}$, or Is It... (Artifact)

Jan de Muijnck-Hughes
University of Glasgow, UK

Wim Vanderbauwhede
University of Glasgow, UK

Abstract

We present two proof-of-concept languages (Circuits & CirQTS) that showcases how *fancy types* (namely linear & dependent types) can enrich hardware design tooling such that we can move existing external static analysis checks into the language's type-system. Using our approach will lead to the enhanced safety of designs, and increase in design productivity, through early identification and reduction of connection errors.

This artifact presents our verified implementations (as realised in Idris2) of the simply (Circuits) and fancily typed (CirQTS) languages, and the test suite used to assess efficacy of our approach.

2012 ACM Subject Classification Software and its engineering → General programming languages; Software and its engineering → Language features; Software and its engineering → Domain specific languages; Software and its engineering → System modeling languages

Keywords and phrases Hardware Design, Linear Types, Dependent Types, DSLs, Idris, SystemVerilog, Netlists

Digital Object Identifier 10.4230/DARTS.9.2.4

Funding The work is funded by EPSRC grants: Border Patrol (EP/N028201/1) and AppControl (EP/V000462/1).

Related Article Jan de Muijnck-Hughes and Wim Vanderbauwhede, "Wiring Circuits Is Easy as $\{0, 1, \omega\}$, or Is It...", in 37th European Conference on Object-Oriented Programming (ECOOP 2023), LIPIcs, Vol. 263, pp. 8:1–8:28, 2023. `https://doi.org/10.4230/LIPIcs.ECOOP.2023.8`

Related Conference 37th European Conference on Object-Oriented Programming (ECOOP 2023), July 17–21, 2023, Seattle, Washington, United States

Evaluation Policy The artifact has been evaluated as described in the ECOOP 2023 Call for Artifacts and the ACM Artifact Review and Badging Policy.

1 Scope

Our work showcases the use of quantitative types in hardware design languages by detailing how we can retrofit quantitative types onto SystemVerilog netlists. Netlists are gate-level descriptions of hardware the are produced as the result of synthesis, it is from these netlists that hardware is generated (fabless or fabbed).

Our paper presents two languages, Circuits & CirQTS. The first, Circuits, presents a simple structural type-system for SystemVerilog netlists that demonstrates how we can type netlists using standard structural techniques and what it means for netlists to be type-safe but still accepted non-linearly wired designs. We then detail how to retrofit the type-system of Circuits (resulting in CirQTS) with quantitative types and make the type-system sub-structural and detail how our new type-safety result ensures that wires and ports are used once.

Dagstuhl Artifacts Series
Schloss Dagstuhl – Leibniz-Zentrum für Informatik,
Dagstuhl Publishing, Germany

Our work presents proof of concept languages that showcases how **fancy types** (namely linear & dependent types) can enrich hardware design tooling such that we can move existing external static analysis checks into the language's type-system. Using our approach will lead to the enhanced safety of designs, and increase in design productivity, through early identification and reduction of connection errors.

This artifact presents our verified implementation of the simply (Circuits) and fancily typed (CirQTS) languages.

2 Content

The artifact package includes:

- `circuits.box`
  - A Virtual Box virtual machine that contains source code & test suites;
- `circuits`
  - A copy of the source code;
- `circuits_doc`
  - A copy of the IdrisDoc;
- `circuits_html`
  - A copy of the katla generated html showing semantically highlighted code;
- `circuits.pdf`
  - A copy of the camera-ready paper submitted to ECOOP 2023;

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the artifact code is also available at: `https://github.com/border-patrol/linear-circuits`.

4 Tested platforms

The artifact has been packaged as a **Virtual Box image** using packer, and has been designed to be run using **Vagrant**.

- `https://www.virtualbox.org/`
- `https://www.vagrantup.com/`

Both Virtual Box and Vagrant are known to work on major platforms.

Alternatively, we have provided the raw sources for the software code along side the virtural machine box. To run these sources you will need a working installation of: **Idris2**, **Verilator**, and **Make**. Details of installing the software code can be found in `INSTALL.md`, located in the top-level directory of the software code.

5 License

The artifact is available under the Clear BSD license.

6 MD5 sum of the artifact

48ffa350d41ecebf1c25e7936a319797

7 Size of the artifact

519 MiB