Vericert OOPSLA 2021 Artifact

This artifact should support the claims made in the paper "Formal Verification of High-Level Synthesis". In the paper, our tool Vericert was referred to as using the temporary name "HLSCert". The claims that can be verified by the paper are the following:

- The mechanised proof of correctness of the translation from C into Verilog is provided and can be checked and rerun.
- All 27 PolyBench benchmarks can be recompiled using Vericert.
- The cycle counts of Vericert on the benchmarks can be checked and compared against LegUp 4.0.
- If Vivado is downloaded separately, then the whole performance section can be checked, including all the graphs that appear in the paper.

Claims that are not supported by the artifact

Unfortunately, we could not include our version of LegUp 4.0 in the artifact due to license restrictions. In addition to that, LegUp was recently bought by Microchip and renamed to SmartHLS[1], which means that it also cannot be freely downloaded anymore either, and the original open source version of LegUp 4.0 is not available anymore due to server issues in Toronto[2]. We have tried contacting the authors of LegUp in Toronto, but have not heard back yet on if our version of LegUp can be shared in the artifact.

Instead, we have included the net lists that LegUp generated from the benchmarks in the artifact, with all the optimisation levels that were tried, however, it does mean that these cannot be verified again and that other optimisation options cannot be tried.

In addition to that, the Vivado synthesis tool by Xilinx[3] is also commercial (but free to download), and therefore cannot be bundled into the artifact either. This synthesis tool was used to get accurate timing information about how the design would run on an FPGA, and also give the area that the design would take up on the FPGA. To be able to reproduce these results, it would therefore need Vivado to be set up so that the scripts can run.

[1] `https://www.microsemi.com/product-directory/fpga-design-tools/5590-hls#software-download`

[2] `https://legup.eecg.utoronto.ca`

[3] `https://www.xilinx.com/support/download.html`

Kick the tyres

First, the docker image needs to be downloaded and run, which contains the git repository:

```
docker pull ymherklotz/vericert
docker run -it ymherklotz/vericert sh
```

Then, one just has to go into the directory which contains the git repository and open a `nix-shell`, which will load a shell with all the correct dependencies loaded:

```
cd /vericert
nix-shell
```

Then, any commands can be run in this shell to run `vericert`, which has already been compiled and can be found in the `/vericert/bin` directory. For a quick test that it is working, a few very simple examples in the `/vericert/test` directory can be run using:

```
make test
```

If this finishes without errors, it means that Vericert is working correctly.

Detailed Artifact Description

This section describes the detailed instructions to get the results for the different sections of the paper, first describing the structure of the proof and how to execute Vericert manually, to finally running Vericert on the benchmarks and get the cycle counts for the Vericert designs as well as the precompiled LegUp designs.

How to manually compile using Vericert

To compile arbitrary C files, the following command can be used:

```
vericert main.c -o main.v
```

Which will generate a Verilog file with a corresponding test bench. The Verilog file can then be simulated by using the Icarus Verilog simulator:

```
iverilog main.v -o main
./main
```

This should print out the return value from the main function in addition to the number of cycles that it took to execute the hardware design.

Getting cycle counts for Vericert

To get the cycle counts for Vericert from the benchmarks, it suffices to compile