FireSim/FPGA - Progress Report

Milan Lagae

Institution/University Name:	Vrije Universiteit Brussel
Faculty:	Sciences and Bioengineering Sciences
Course:	Master Thesis

Contents

1 Introduction

The list of resources can be found in the section §2. First trying to run using the FireSim tool in section §3.1 and the last section on executing manual commands in section §3.5.

2 Resources

The following list of resources where used during the course of this research, the most important are listed below:

- …

/

3 Metasimulation

This is about getting metasimulation working in FireSim using the open source simulator Verilator.

3.1 FireSim & Verilator

3.2 Installation

Installing can be done by following the tutorials on the firesim & chipyard documentation website, installation will take some time and the host platform is required to be a Linux x86 machine for easy of support.

3.3 Config

Firesim works based on the use of *.yaml configuration scripts. For the metasimulation use case the configuration files that are important are the following two:

- **ADD**
- **ADD**

These files can be copied from the examples directory in the chipyard repo.

The config_runtime.yaml file should be modified with the following section of code:

INSERT CODE

The metasimulation will be done on the same hosts we are managing firesim from. This configuration for the metasimulation slots must be specified in the following file: **ADD**. And can than be reference in the config_runtime.yaml file.

For each run configuration a chip configuration must be specified, the most important value of the config, is the name of the config in the config directory.

This name must match the name of the config defined, which specifies the numbers of cores,memory, abstract config, .. .

Due to configuration/setup at home some small modifications had to be made to the python file executing the firesim commands, this file is named: **ADD** and can be found in: **ADD**.

3.3.1 Digression:

From the method: **ADD** remove the annotation **ADD** and add the following code in the method, looking for an vironemnt variable called: **ADD**, which must be specified for each command that is run, and than set, so the ssh Fabric package does not complain about sudo/parallism requiring a password.

3.4 Setup & Workload

After everything is setup, the firesim infrasetup command can be run, with the first example gcd there should be no errors, and the following kind of output should be visible:

ADD

If that is the case the connection is successful. Now we modify the config to execute a named config from the config directory. Change the **ADD** file to the following:

ADD

If that is done, we can run the infrasetup command again.

FROM this point, file not found exception

Contininuing from this point, running the config, on my end I got the java file not found exception. No solution has been found at the moment of writing this.

3.5 Chipyard & Verilator

Since FireSim makes use of Verilator under the hood, I decided to perform manual command executions.

We will make use of the config we specified earlier. Proceed to the sims/verilator directory in the chipyard repository. Executing the following command to compile your config: make CONFIG=<config-name>. This command should execute without a problem. An example of this command can be found below:

ADD IMAGE

After this we can proceed to run a binary, before we can run an RISC-V binary, we must first compile a binary to run. Go into the test/ directory inside the Chipyard repository. You can decide which binary you want to run, to compile assembly code, read the README.md file inside of the directory, it describes on how to compiler all/or a specific binary. I will assume you compiled the mt-hello.o file to a mt-hello.riscv binary.

With the binary compiled proceed to execute the following command: make CONFIG=<config-name> BINARY=<path-to-binary> run-binary-hex

The last run-binary-hex specifies to run the binary using the fastmem option, more infor is available in the documentary. Specifying the fastmem option, noticably increases the speed of the simulation. When executing the mt-hello.riscv binary, the output will look like something below:

ADD IMAGE

4 Custom Tile/Cores

TODO