FireSim

A Brief Tour of FireSim: The Manager & Compiler; Building Hardware Designs

https://fires.im

@firesimproject

Micro Tutorial 2024

Speaker: Joonho Whangbo
(Original slides by Abraham Gonzalez)

Berkeley Architecture Research

Agenda: What Will We Cover?

1) The Compiler → Golden Gate

- Invoke it on example RTL
- Inspect its outputs

2) The Manager → `firesim`

- Explain how it's configured
- Demonstrate how it's used to build bitstreams

Where is FireSim in Chipyard?

With the software RTL simulators!

`~/chipyard-afternoon/sims/firesim`

→ This has been exported as `$FDIR`

Interactive:

```
# <open new terminal to ec2 instance>

$ tmux new -s afternoon

$ cd $FDIR

$ source sourceme-manager.sh

$ ls
```

FireSim's Directory Structure

`sim/`

- Golden Gate lives here
- Scala & C++ sources for additional FireSim models
- Make-based build system to invoke Golden Gate

`deploy/`

- Manager lives here
- FireSim workload definitions

`platforms/` → FPGA platform definitions (e.g. AWS FPGA for F1, Xilinx Vitis for U250)

`sw/` → target software & FireMarshal (more on this later)

Agenda: What Will We Cover?

1) The Compiler → “Golden Gate”

- Invoke it on example RTL
- Inspect its outputs

2) The Manager → `firesim`

- Explain how it’s configured
- Demonstrate how it’s used to build bitstreams

An Analogy

- Golden Gate is like Verilator but for FPGA-accelerated simulation

Verilator generates C++ sources to simulate your design.

→ Compile and run on a CPU-host

Golden Gate generates C++ & Verilog to simulate your design.

→ Compile and run on a hybrid CPU & FPGA host

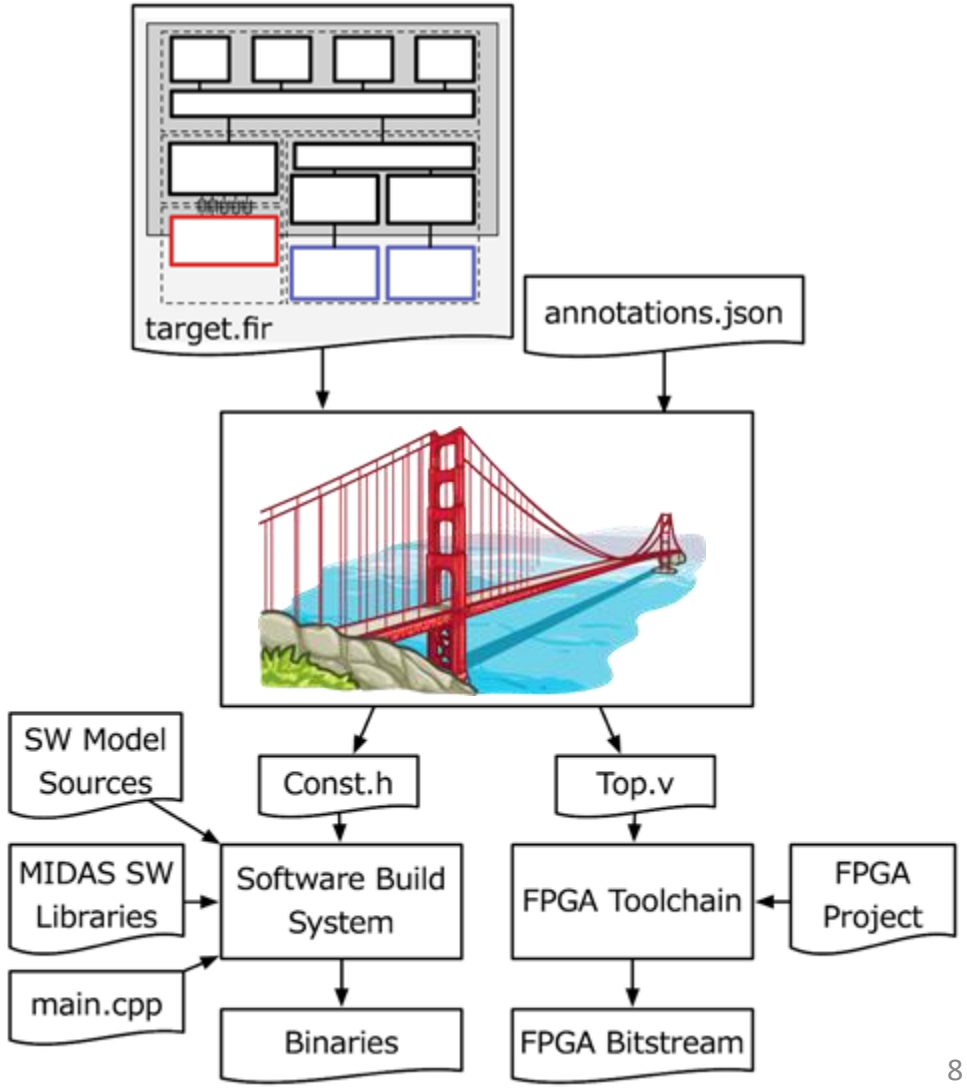
Golden Gate Compiler

Inputs:

- FIRRTL & annos from a Chipyard generator
- Compiler configuration

→ Produces sources for a simulator that are:

- deterministic
- support co-simulation of software models
- *area-optimized to fit more on the FPGA*



Interactive:

```
$ cd $FDIR/sim/generated-src/f1

$ ls

# here you'll find output directories for all builds

$ cd <any-directory-here>

$ ls
```

Inspecting the Outputs

`<long-name>.fir & <long-name>.anno.json`

- Target's FIRRTL & annotations

`FireSim-generated.sv`

- The compiled simulator

`FireSim-generated.const.h`

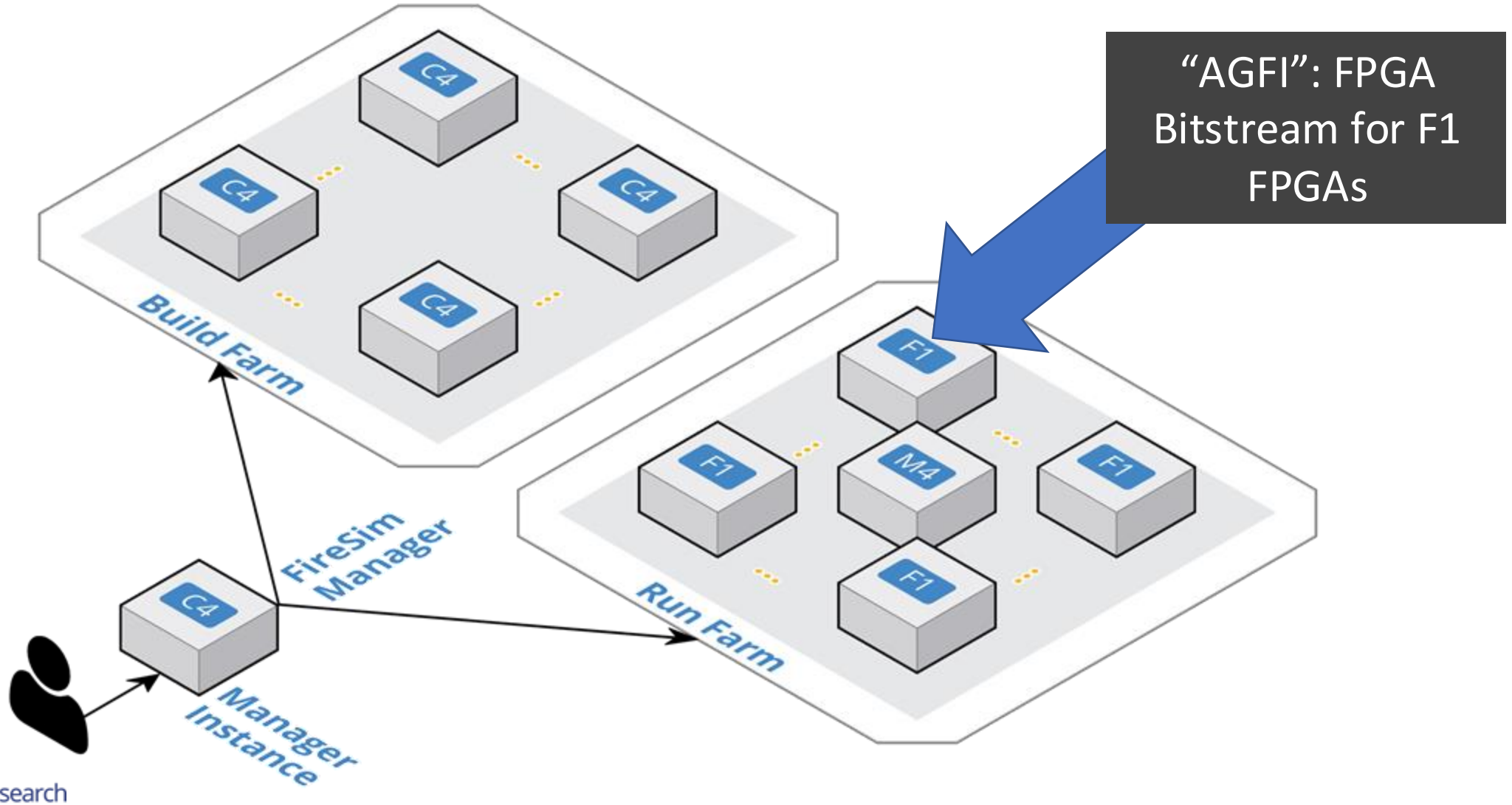
- Simulator's memory map

Agenda: What Will We Cover?

1) The Compiler → Golden Gate
- Invoke it on example RTL
- Inspect its outputs

2) The Manager → `firesim`
- Explain how it's configured
- Demonstrate how it's used to build bitstreams

Background Terminology



Using the `firesim` Manager Command Line

- Sourcing `sourceme-manager.sh` puts `firesim` on your path
  - Can call `firesim` from anywhere on the instance
  - It will always run from the directory:

`$FDIR/deploy/`

After a fresh clone, need to call:

```
firesim managerinit --platform f1
```

→ **You already did this at the start of the tutorial**

Interactive:

```
$ cd $FDIR/deploy

$ ls
```

Configuring the Manager. 4 files in `firesim/deploy/`

config_build.yaml

config_build_recipes.yaml

```yaml
# Build-time build recipe configuration for the FireSim Simulation Manager
# See https://docs.fires.im/en/stable/Advanced-Usage/Manager/Manager-Config

# this file contains sections that describe hardware designs that /can/ be
# edit config_build.yaml to actually "turn on" a config to be built when yo
# buildbitstream

##########
# Schema:
##########
# <NAME>:
#     DESIGN: <>
#     TARGET_CONFIG: <>
#     PLATFORM_CONFIG: Config
#     deploy_quintuplet: null
#     # NOTE: these platform_config_args are for F1 only
#     # they should be set to null if using another platform
#     platform_config_args:
#         fpga_frequency: null
#         build_strategy: null
#     post_build_hook: null
#     metasim_customruntimeconfig: "path to custom runtime config for metasi
#     bit_builder_recipe:
#     # OPTIONAL: overrides for bit builder recipe
#     # Arg structure should be identical to the args given
#     # in the base_recipe.
#     #bit_builder_arg_overrides:
#     #   <ARG>: <OVERRIDE>

# Quad-core, Rocket-based recipes
# REQUIRED FOR TUTORIALS
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
    PLATFORM: f1
    TARGET_PROJECT: firesim
    DESIGN: FireSim
    TARGET_CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithF
    PLATFORM_CONFIG: WithAutoILA_BaseF1Config
    deploy_quintuplet: null
    platform_config_args:
        fpga_frequency: 90
        build_strategy: TIMING
    post_build_hook: null
    metasim_customruntimeconfig: null
    bit_builder_recipe: bit-builder-recipes/f1.yaml

# NB: This has a faster host-clock frequency than the NIC-based design, bec
# its uncore runs at half rate relative to the tile.
firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3:
    PLATFORM: f1
    TARGET_PROJECT: firesim
    DESIGN: FireSim
    TARGET_CONFIG: DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSimTe
    PLATFORM_CONFIG: WithAutoILA_BaseF1Config
    deploy_quintuplet: null
    platform_config_args:
        fpga_frequency: 140
        build_strategy: TIMING
    post_build_hook: null
    metasim_customruntimeconfig: null
    bit_builder_recipe: bit-builder-recipes/f1.yaml
```

config_hwdb.yaml

```yaml
# Hardware config database for FireSim Simulation Manager
# See https://docs.fires.im/en/stable/Advanced-Usage/Manager.

# Hardware configs represent a combination of an agfi, a dep
# (if needed), and a custom runtime config (if needed)

# The AGFIs provided below are public and available to all u
# Only AGFIs for the latest release of FireSim are guarantee
# If you are using an older version of FireSim, you will nee
# own images.

# DOCREF START: Example HWDB Entry
firesim_boom_singlecore_nic_l2_llc4mb_ddr3:
    agfi: agfi-0aac270576e64693c
    deploy_quintuplet_override: null
    custom_runtime_config: null
# DOCREF END: Example HWDB Entry
firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3:
    agfi: agfi-02f92e7c011ef6e19
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_gemmini_printf_rocket_singlecore_no_nic:
    agfi: agfi-0ace16d35c5758893
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_gemmini_rocket_singlecore_no_nic:
    agfi: agfi-05eec5fb565f7cfa3
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
    agfi: agfi-0455e4c2892076c1a
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3:
    agfi: agfi-09eeb63f4fae0929e
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_rocket_singlecore_sha3_nic_l2_llc4mb_ddr3:
    agfi: agfi-02e4056f9bec5a240
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3:
    agfi: agfi-0d8abef077c23a4de
    deploy_quintuplet_override: null
    custom_runtime_config: null
firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3_printf:
    agfi: agfi-033e840230f51668f
    deploy_quintuplet_override: null
    custom_runtime_config: null
```

config_runtime.yaml

Configuring a Build

config_build.yaml

```yaml
# Build-time build design / AGFI configuration for the FireSim
# See https://docs.fires.im/en/stable/Advanced-Usage/Manager/M

# this refers to build farms defined in config_build_farm.yaml
build_farm:
  base_recipe: build-farm-recipes/aws_ec2.yaml
  recipe_arg_overrides:
    # tag to apply to build farm hosts
    build_farm_tag: mainbuildfarm
    # instance type to use per build
    instance_type: z1d.2xlarge
    # instance market to use per build (ondemand, spot)
    build_instance_market: ondemand
    # if using spot instances, determine the interrupt behavio
    spot_interruption_behavior: terminate
    # if using spot instances, determine the max price
    spot_max_price: ondemand
    # default location of build directory on build host
    default_build_dir: /home/centos/firesim-build

builds_to_run:
    # this section references builds defined in config_build_r
    # if you add a build here, it will be built when you run b

    # Unnetworked designs use a three-domain configuration
    # Tiles: 1000 MHz
    #    <Rational Crossing>
    # Uncore: 500 MHz
    #    <Async Crossing>
    # DRAM : 1000 MHz
    - firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3
    - firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3

    # All NIC-based designs use the legacy FireSim frequency s
    # tiles and uncore running at 3.2 GHz to sustain 200Gb the
```

config_build_recipes.yaml

```yaml
# Build-time build recipe configuration for the FireSim Simulation Manager
# See https://docs.fires.im/en/stable/Advanced-Usage/Manager/Manager-Configura

# this file contains sections that describe hardware designs that /can/ be bui
# edit config_build.yaml to actually "turn on" a config to be built when you r
# buildbitstream

###########
# Schema:
###########
# <NAME>:
#    DESIGN: <>
#    TARGET_CONFIG: <>
#    PLATFORM_CONFIG: Config
#    deploy_quintuplet: null
#    # NOTE: these platform_config_args are for F1 only
#    # they should be set to null if using another platform
#    platform_config_args:
#       fpga_frequency: null
#       build_strategy: null
#    post_build_hook: null
#    metasim_customruntimeconfig: "path to custom runtime config for metasims"
#    bit_builder_recipe:
#    # OPTIONAL: overrides for bit builder recipe
#    # Arg structure should be identical to the args given
#    # in the base_recipe.
#    #bit_builder_arg_overrides:
#    #  <ARG>: <OVERRIDE>

# Quad-core, Rocket-based recipes
# REQUIRED FOR TUTORIALS
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
    PLATFORM: f1
    TARGET_PROJECT: firesim
    DESIGN: FireSim
    TARGET_CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFire
    PLATFORM_CONFIG: WithAutoILA_BaseF1Config
    deploy_quintuplet: null
    platform_config_args:
        fpga_frequency: 90
        build_strategy: TIMING
    post_build_hook: null
    metasim_customruntimeconfig: null
    bit_builder_recipe: bit-builder-recipes/f1.yaml
```

Anatomy of a Build Recipe

config_build_recipes.yaml

```yaml
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
    PLATFORM: f1
    TARGET_PROJECT: firesim
    DESIGN: FireSim
    TARGET_CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFi
    PLATFORM_CONFIG: WithAutoILA_BaseF1Config
    deploy_quintuplet: null
    platform_config_args:
        fpga_frequency: 90
        build_strategy: TIMING
    post_build_hook: null
    metasim_customruntimeconfig: null
    bit_builder_recipe: bit-builder-recipes/f1.yaml
```

```
WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_Wi
thFireSimHighPerfConfigTweaks_chipyard.QuadRocketConf
ig
```

Consists of:

- A label
- The tuple (DESIGN, TARGET_CONFIG, PLATFORM_CONFIG)
- Platform-specific bitstream generation parameters

Defining a Build Job: `config_build.yaml`

```
build_farm:
  # managerinit replace start
  base_recipe: build-farm-recipes/aws_ec2.yaml
  # Uncomment and add args to override defaults.
  # Arg structure should be identical to the args given
  # in the base_recipe.
  #recipe_arg_overrides:
  #   <ARG>: <OVERRIDE>
  # managerinit replace end

builds_to_run:
    # this section references builds defined in config_build_r
    # if you add a build here, it will be built when you run b

    # Unnetworked designs use a three-domain configuration
    # Tiles: 1600 MHz
    #     <Rational Crossing>
    # Uncore: 800 MHz
    #     <Async Crossing>
    # DRAM : 1000 MHz
    - firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3
    - firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3

    # All NIC-based designs use the legacy FireSim frequency s
    # tiles and uncore running at 3.2 GHz to sustain 200Gb the
    - firesim_supernode_rocket_singlecore_nic_l2_lbp
    - firesim_rocket_quadcore_nic_l2_llc4mb_ddr3
```

Consists of:

- Build host platform configuration
- A list of recipes you'd like to batch out to a build farm

Defining a Build Job: `config_build.yaml`

```
    - firesim_rocket_quadcore_nic_l2_llc4mb_ddr3
    - firesim_boom_singlecore_nic_l2_llc4mb_ddr3

    # Configs for tutorials
    # - firesim_rocket_singlecore_no_nic_l2_lbp
    # - firesim_rocket_singlecore_sha3_nic_l2_llc4mb_ddr3
    # - firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3
    # - firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3_p

agfis_to_share:
    - firesim_rocket_quadcore_nic_l2_llc4mb_ddr3
    - firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3
    - firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3
    - firesim_boom_singlecore_nic_l2_llc4mb_ddr3

    - firesim_supernode_rocket_singlecore_nic_l2_lbp

    # Configs for tutorials
    # - firesim_rocket_singlecore_no_nic_l2_lbp
    # - firesim_rocket_singlecore_sha3_nic_l2_llc4mb_ddr3
    # - firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3
    # - firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3_p

share_with_accounts:
    # To share with a specific user:
    somebodysname: 123456789012
    # To share publicly:
    #public: public
```

Once you're done with builds:

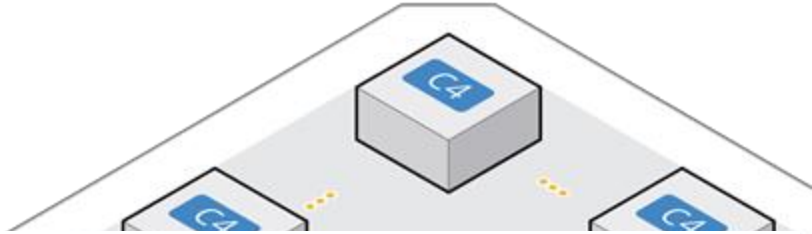
- A list of recipes you'd like to share with other users

Running builds

- Once we've configured ***what*** we want to build, let's build it

```
$ firesim buildbitstream
```

- This completely automates the process. For each design, in-parallel:
  - Launch a build instance
  - Generate target RTL & invokes Golden Gate
  - Ship infrastructure to build instances, run Vivado FPGA builds in parallel
  - Collect results back onto manager instance
    - `$FDIR/deploy/results-build/<TIMESTAMP>-<tuple>/`
  - Email you the entry to put into `config_hwdb.yaml`
  - Terminate the build instance



AWS Notifications <no-reply@sns.amazonaws.com>
to me

Your AGFI has been created!
Add

```
firesim_rocket_singlecore_no_nic_l2_lbp:
   agfi: agfi-0e27eb94672e2f5a9
   deploy_triplet_override: null
   custom_runtime_config: null
```

to your config_hwdb.yaml to use this hardware configuration.

Anatomy of a HWDB Entry

```
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
    agfi: agfi-0c45d995a46cce5dc
    deploy_triplet_override: null
    custom_runtime_config: null
```

- Same label as before
- The FPGA image

Hooks to change:

- Software models
- Runtime arguments

→ *Without FPGA recompilation*

Summary

- Don’t fret if you didn’t catch everything, everything we showed you today is documented in excruciating detail at https://docs.fires.im
- We learned how to:
  - Build FireSim FPGA images for a set of targets
    - https://docs.fires.im/en/stable/Building-a-FireSim-AFI.html