



# FireSim

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

<https://fires.im>



@firesimproject

**ASPLOS Tutorial 2023**

Speaker: 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:

```
# <ssh back onto your ec2 instance>
```

```
$ tmux new -s afternoon
```

```
$ cd $FDIR
```

```
$ 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



# Interactive:

```
$ cd $FDIR/sim
```

```
$ make DESIGN=FireSim
```



# 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





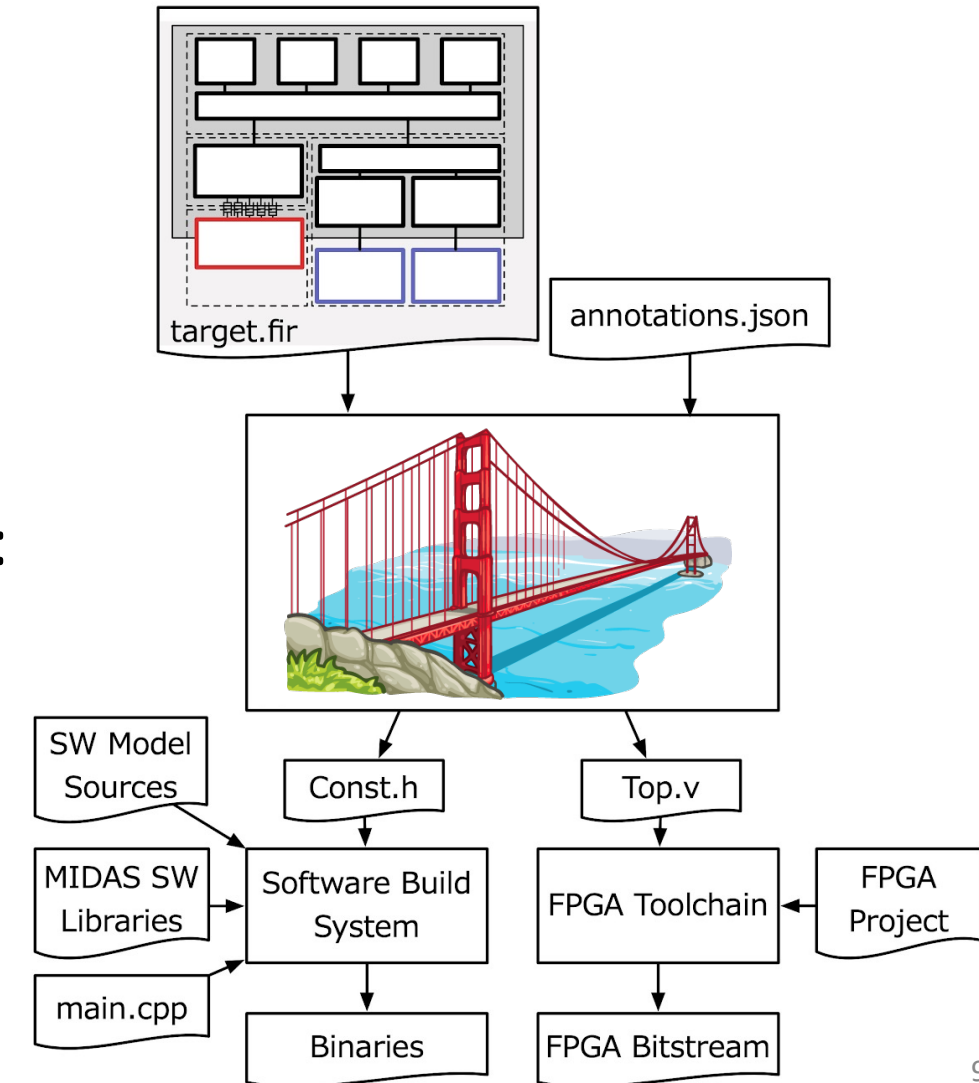
# Golden Gate Compiler

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*





# Interacting with Golden Gate via Make

- Make invokes Golden Gate with three variables (the “Tuple”):

DESIGN :

- The top-level module → MODEL in Chipyard

TARGET\_CONFIG:

- The generator’s config → CONFIG in Chipyard

PLATFORM\_CONFIG:

- Compiler options passed to Golden Gate



# Interactive:

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

```
$ 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

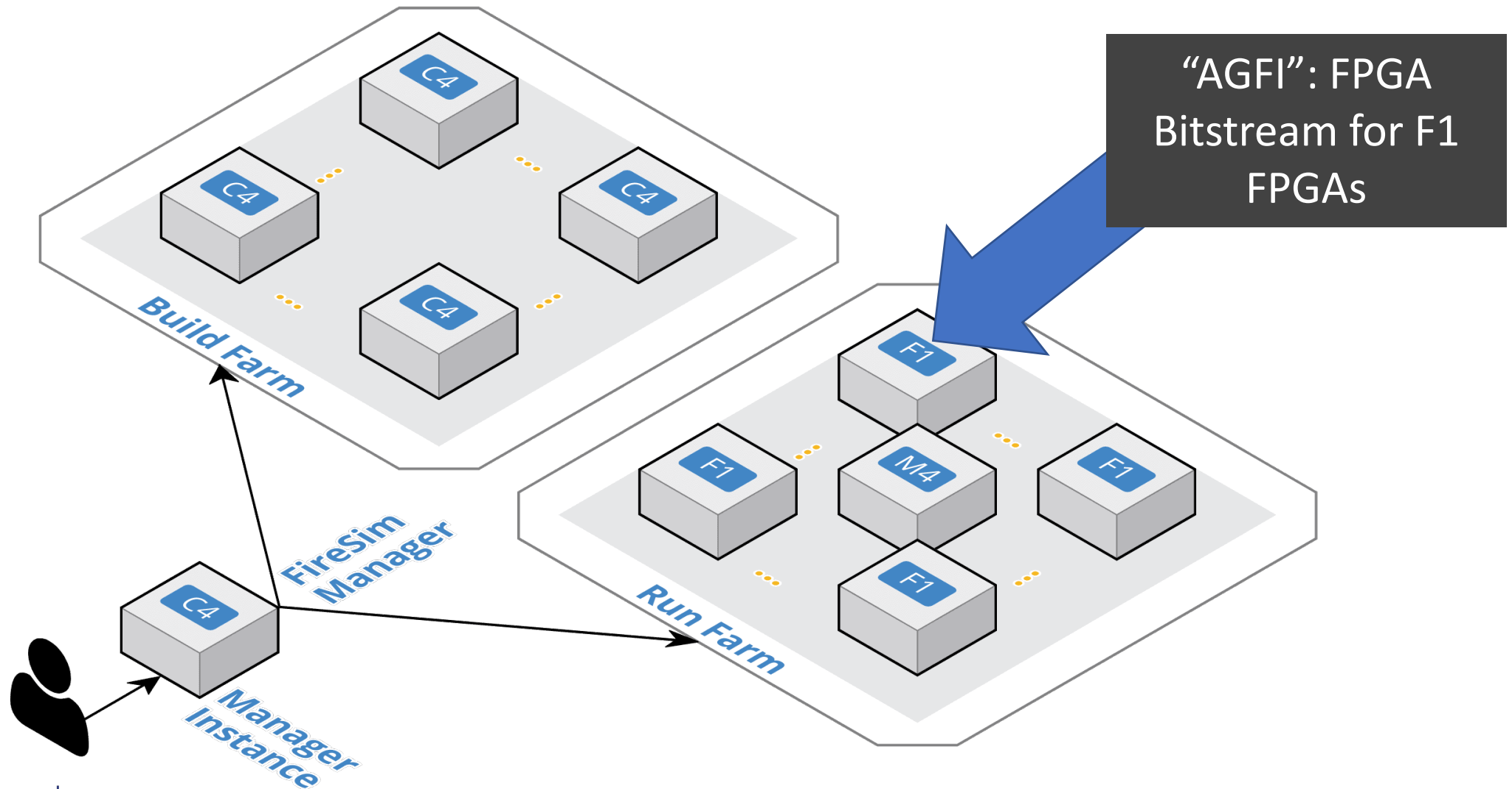
- Invoke it on example RTL
- Inspect its outputs

## 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-f1-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

```
# Build-time build design / AGFI configuration for the FireSim
# See https://docs.firesim/en/stable/Advanced-Usage/Manager/Manager-Config.html

# this refers to build farms defined in config_build_farm.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.yaml
  # if you add a build here, it will be built when you run build

  # 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
  # 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
  - 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
```

## config\_build\_recipes.yaml

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

# 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
# buildfarm:

#####
# Schema:
#####
# <NAME>:
# DESIGN: <>
# TARGET CONFIG: <>
# PLATFORM CONFIG: Config
# deploy triplet: null
# post build hook: null
# metasim_customruntimeconfig: "path to custom runtime config for metasim"
# bit_builder_recipe:
# # OPTIONAL: overrides for bit builder recipe
# # Arg structure should be identical to the args given
# # in the base recipe.
# #bit_builder_recipe overrides:
# # <ARG>: <OVERRIDE>

# Quad-core, Rocket-based recipes
# REQUIRED FOR TUTORIALS
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
  DESIGN: FireSim
  TARGET CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSim
  PLATFORM CONFIG: WithAutoILA_F90MHz_BaseF1Config
  deploy triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/fl.yaml

# NB: This has a faster host-clock frequency than the NIC-based design, but
# its uncore runs at half rate relative to the tile.
firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3:
  DESIGN: FireSim
  TARGET CONFIG: DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSim
  PLATFORM CONFIG: WithAutoILA_F140MHz_BaseF1Config
  deploy triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/fl.yaml

# Single-core, BOOM-based recipes
# REQUIRED FOR TUTORIALS
firesim_boom_singlecore_nic_l2_llc4mb_ddr3:
  DESIGN: FireSim
  TARGET CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSim
  PLATFORM CONFIG: WithAutoILA_F65MHz_BaseF1Config
  deploy triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/fl.yaml

# NB: This has a faster host-clock frequency than the NIC-based design, but
# its uncore runs at half rate relative to the tile.
firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3:
  DESIGN: FireSim
  TARGET CONFIG: DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSim
  PLATFORM CONFIG: WithAutoILA_F65MHz_BaseF1Config
  deploy triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/fl.yaml
```

## config\_hwdb.yaml

```
# Hardware config database for FireSim Simulation Manager
# See https://docs.firesim/en/stable/Advanced-Usage/Manager/Manager-Config.html

# 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 users
# Only AGFIs for the latest release of FireSim are guaranteed
# If you are using an older version of FireSim, you will need
# own images.

firesim_boom_singlecore_nic_l2_llc4mb_ddr3:
  agfi: agfi-0b969bdcc09663973
  deploy triplet override: null
  custom_runtime_config: null

firesim_boom_singlecore_no_nic_l2_llc4mb_ddr3:
  agfi: agfi-0d9d8e9255c80dac5
  deploy triplet override: null
  custom_runtime_config: null

# DOREF START: Example HWDB Entry
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
  agfi: agfi-0c45d995a46ce5dc
  deploy triplet override: null
  custom_runtime_config: null
# DOREF END: Example HWDB Entry

firesim_rocket_quadcore_no_nic_l2_llc4mb_ddr3:
  agfi: agfi-08719c613c2f314cc
  deploy triplet override: null
  custom_runtime_config: null

firesim_supernode_rocket_singlecore_nic_l2_lbp:
  agfi: agfi-0b747b88806aed5c
  deploy triplet override: null
  custom_runtime_config: null

firesim_rocket_singlecore_no_nic_l2_lbp:
  agfi: agfi-0e27eb94672e2f5a9
  deploy triplet override: null
  custom_runtime_config: null

firesim_rocket_singlecore_sha3_nic_l2_llc4mb_ddr3:
  agfi: agfi-064592b4699c8b4d4
  deploy triplet override: null
  custom_runtime_config: null

firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3:
  agfi: agfi-0d668a1f6883c7625
  deploy triplet override: null
  custom_runtime_config: null

firesim_rocket_singlecore_sha3_no_nic_l2_llc4mb_ddr3_printf:
  agfi: agfi-0b811d62bd5294f45
  deploy triplet override: null
  custom_runtime_config: null
```

## config\_runtime.yaml

```
# RUNTIME configuration for the FireSim Simulation Manager
# See https://docs.firesim/en/stable/Advanced-Usage/Manager/Manager-Config.html

run_farm:
  # managerinit replace start
  base_recipe: run-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

metasimulation:
  metasimulation_enabled: false
  # vcs or verilator, use vcs:debug or verilator:debug for waveform generation
  metasimulation_host_simulator: verilator
  # plusargs passed to the simulator for all metasimulations
  metasimulation_only_plusargs: "+fesvr-step-size=128 +dramsim +max-cyc
  # plusargs passed to the simulator ONLY FOR vcs metasimulations
  metasimulation_only_vcs_plusargs: "+vcs+initreg+0 +vcs+initmem+0"

target_config:
  # Set topology: no net config to run without a network simulation
  topology: example_8config
  no_net_num_nodes: 2
  link_latency: 6405
  switching_latency: 10
  net_bandwidth: 200
  profile_interval: -1

  # This references a section from config_hwdb.yaml for fpga-accelerators
  # or from config_build_recipes.yaml for metasimulation
  # In homogeneous configurations, use this to set the hardware config
  # for all simulators
  default_hw_config: firesim_rocket_quadcore_nic_l2_llc4mb_ddr3

  # Advanced: Specify any extra plusargs you would like to provide when
  # booting the simulator (in both FPGA-sim and metasim modes). This
  # is a string, with the contents formatted as if you were passing the
  # at command line, e.g. "+a=1 +b=2"
  plusarg_passthrough: ""

tracing:
  enable: no

  # Trace output formats. Only enabled if "enable" is set to "yes" at
  # 0 = human readable; 1 = binary (compressed raw data); 2 = flamegraph
  # unwinding -> Flame Graph)
  output_format: 0

  # Trigger selector.
  # 0 = no trigger; 1 = cycle count trigger; 2 = program counter trigger
  # instruction trigger
  selector: 1
  start: 0
  end: -1

autocounter:
  read_rate: 0

workload:
  workload_name: linux-uniform.json
  terminate_on_completion: no
  suffix_tag: null

host_debug:
  # When enabled (=yes), Zeros-out FPGA-attached DRAM before simulation
```





# Configuring a Build

## config\_build.yaml

```
# Build-time build design / AGFI configuration for the FireSim
# See https://docs.firesim/en/stable/Advanced-Usage/Manager/Manager-Config.html

# this refers to build farms defined in config_build_farm.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_recipe.yaml
  # if you add a build here, it will be built when you run build

  # 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
```

## config\_build\_recipes.yaml

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

# 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
# buildafi

#####
# Schema:
#####
# <NAME>:
#   DESIGN: <>
#   TARGET_CONFIG: <>
#   PLATFORM_CONFIG: Config
#   deploy_triplet: null
#   post_build_hook: null
#   metasim_customruntimeconfig: "path to custom runtime config for meta
#   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:
  DESIGN: FireSim
  TARGET_CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithNIC
  PLATFORM_CONFIG: WithAutoILA_F90MHz_BaseF1Config
  deploy_triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/f1.yaml
```



# Anatomy of a Build Recipe

`config_build_recipes.yaml`

Consists of:

```
firesim_rocket_quadcore_nic_l2_llc4mb_ddr3:
  DESIGN: FireSim
  TARGET_CONFIG: WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFi
  PLATFORM_CONFIG: WithAutoILA_F90MHz_BaseF1Config
  deploy_triplet: null
  post_build_hook: null
  metasim_customruntimeconfig: null
  bit_builder_recipe: bit-builder-recipes/f1.yaml
```

- A label
- The tuple from before
- 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



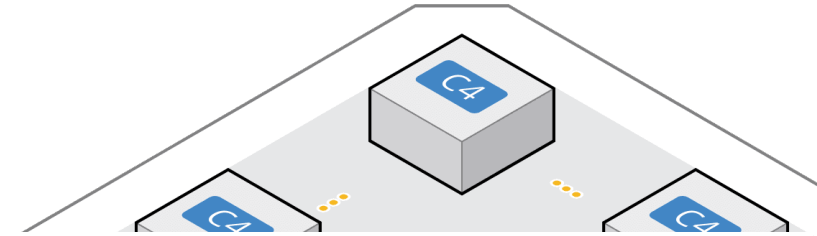


# Running builds

- 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

- Same label as before
- The FPGA image

Hooks to change:

- Software models
- Runtime arguments

→ *Without FPGA recompilation*

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



# 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>