



FireSim

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

<https://firesim.com>



@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 source-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



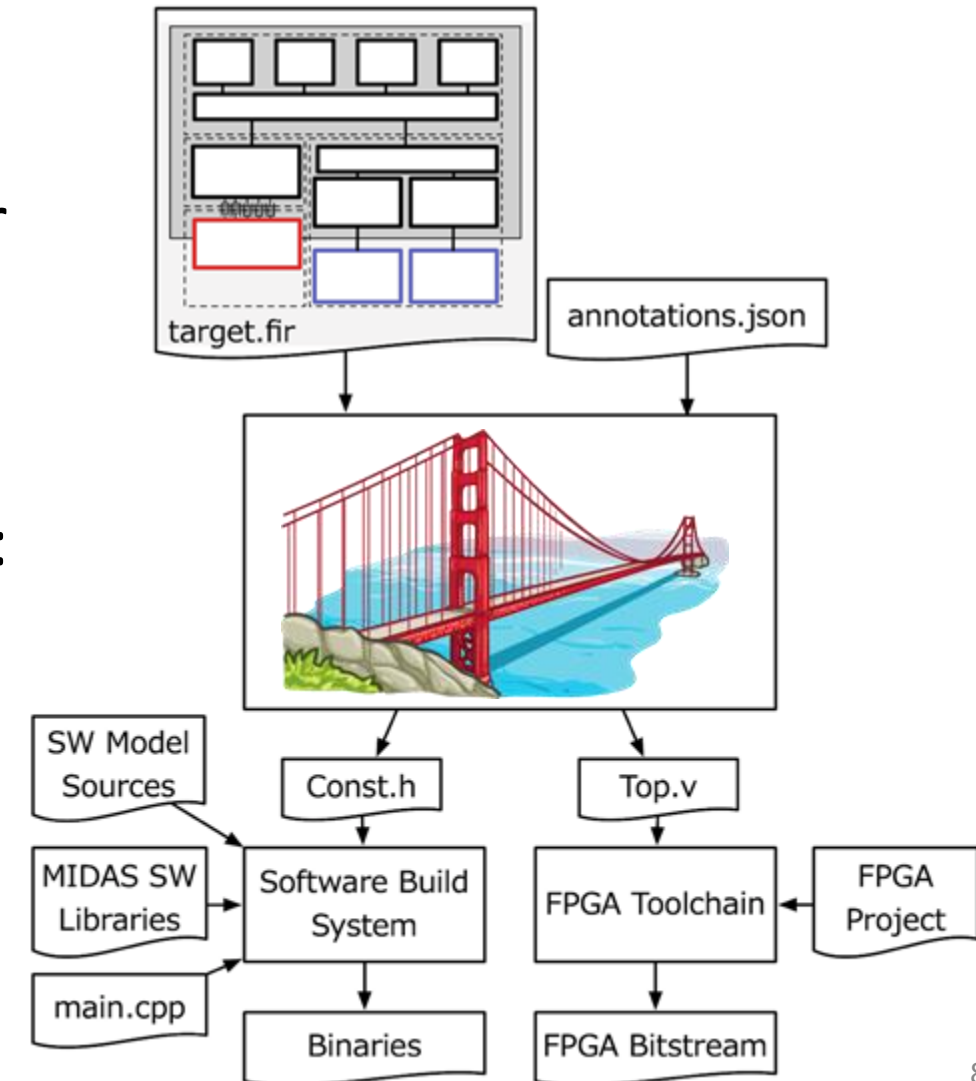
Golden Gate Compiler

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

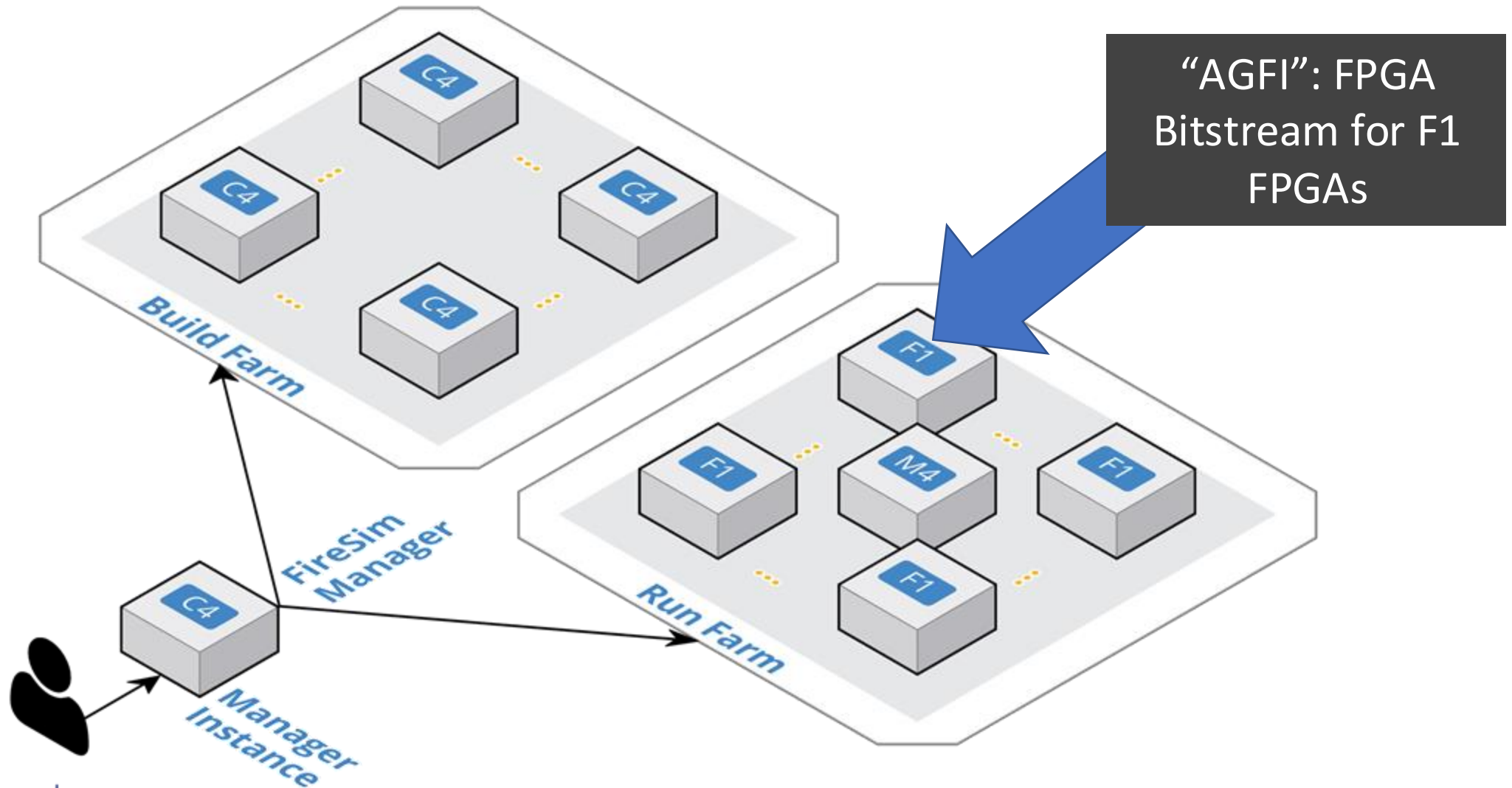
- Invoke it on example RTL
- Inspect its outputs

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 `source-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_farm:
  base_recipe: build-farm-recipes/awr_ec2.yaml
  recipe_arg_override:
    # tag to apply to build farm hosts
    build_farm_tag: awsbuildfarm
    # instance type to use per build
    instance_type: r5d.2xlarge
    # instance market to use per build (ondemand, spot)
    build_instance_market: ondemand
    # If using spot instances, determine the interrupt behavior (terminate, stop, hibernates)
    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_recipes.yaml
  # if you add a build here, it will be built when you run buildbitstream

  # Uncommented designs use a three-domain configuration
  # Tiles: 1000 MHz
  # - optional1_Crossings
  # - Uncore: 500 MHz
  # - <Async Crossings>
  # - DMM: 1000 MHz
  - firesim_rocket_quadcore_no_nic_12_1lc4mb_ddr3
  - firesim_boom_singlecore_no_nic_12_1lc4mb_ddr3

  # All NIC-based designs use the legacy FireSim frequency selection, with the
  # Tiles and uncore running at 3.2 GHz to sustain 20000 theoretical NIC BW
  - firesim_supermode_rocket_singlecore_nic_12_1lc4mb_ddr3
  - firesim_rocket_quadcore_nic_12_1lc4mb_ddr3
  - firesim_boom_singlecore_nic_12_1lc4mb_ddr3

  # Configs for tutorials
  - firesim_rocket_singlecore_no_nic_12_1lc4mb_ddr3
  - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3
  - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3
  - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3_printf
  - firesim_gemini_rocket_singlecore_no_nic
  - firesim_gemini_printf_rocket_singlecore_no_nic

  # Config for Vitis/ZVM
  # - vitis_firesim_rocket_singlecore_no_nic
  # - vitis_firesim_gemini_rocket_singlecore_no_nic

  # Config for Xilinx Alveo U200/U280
  # - alveo_u200_firesim_rocket_singlecore_no_nic
  # - alveo_u280_firesim_rocket_singlecore_no_nic

  # Config for Xilinx VCU118
  # - xilinx_vcu118_firesim_rocket_singlecore_400_no_nic

  # Config for @H0Research NitroFury II
  # - nitroFury_firesim_rocket_singlecore_no_nic

agfi_to_share:
  - firesim_rocket_quadcore_nic_12_1lc4mb_ddr3
  - firesim_rocket_quadcore_no_nic_12_1lc4mb_ddr3
  - firesim_boom_singlecore_no_nic_12_1lc4mb_ddr3
  - firesim_boom_singlecore_nic_12_1lc4mb_ddr3
  - firesim_supermode_rocket_singlecore_nic_12_1lc4mb_ddr3

  # Configs for tutorials
  # - firesim_rocket_singlecore_no_nic_12_1lc4mb_ddr3
  # - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3
  # - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3
  # - firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3_printf

  # Config for @H0Research NitroFury II
  # - nitroFury_firesim_rocket_singlecore_no_nic

share_with_accounts:
  # To share with a specific user:
  # user: centos:1334679912
  # 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

# 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 you
# 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 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_arg_overrides:
# <ARG>: <OVERRIDE>

# Quad-core, Rocket-based recipes
# REQUIRED FOR TUTORIALS
firesim_rocket_quadcore_nic_12_1lc4mb_ddr3:
  PLATFORM: f1
  TARGET_PROJECT: firesim
  DESIGN: FireSim
  TARGET_CONFIG: WithNIC_DDR3FRFCPLL4MB_WithDefaultFireSimBridges_With
  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_12_1lc4mb_ddr3:
  PLATFORM: f1
  TARGET_PROJECT: firesim
  DESIGN: FireSim
  TARGET_CONFIG: DDR3FRFCPLL4MB_WithDefaultFireSimBridges_WithFireSimT
  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

```

# Hardware config database for FireSim Simulation Manager
# See https://docs.firesim/en/stable/Advanced-Usage/Manager/Manager-Configuration-4

# 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 need
# own images.

# DOCREF START: Example HWDB Entry
firesim_boom_singlecore_nic_12_1lc4mb_ddr3:
  agfi: agfi-0aac270576e64693c
  deploy Quintuplet_override: null
  custom_runtime_config: null
# DOCREF END: Example HWDB Entry
firesim_boom_singlecore_no_nic_12_1lc4mb_ddr3:
  agfi: agfi-02f92e7c011ef6e19
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_gemini_printf_rocket_singlecore_no_nic:
  agfi: agfi-0ace16d35c5758893
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_gemini_rocket_singlecore_no_nic:
  agfi: agfi-05eec5fb565f7cfa3
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_rocket_quadcore_nic_12_1lc4mb_ddr3:
  agfi: agfi-0455e4c2892076c1a
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_rocket_quadcore_no_nic_12_1lc4mb_ddr3:
  agfi: agfi-09eeb63f4fae0929e
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_rocket_singlecore_sha3_nic_12_1lc4mb_ddr3:
  agfi: agfi-02e4056f9bec5a240
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_rocket_singlecore_sha3_no_nic_12_1lc4mb_ddr3:
  agfi: agfi-0d8abef077c23a4de
  deploy Quintuplet_override: null
  custom_runtime_config: null
firesim_rocket_singlecore_sha3_no_nic_12_1lc4mb_ddr3_printf:
  agfi: agfi-033e840230f51668f
  deploy Quintuplet_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-Configuration-4

run_farm:
  base_recipe: run-farm-recipes/awr_ec2.yaml
  recipe_arg_override:
    # tag to apply to run farm hosts
    run_farm_tag: awsbuildfarm
    # enable expanding run farm by run_farm_hosts given
    # class. ondemand, spot, f1m
    # minutes to wait attempting to request instances
    launch_instance_timeout_minutes: 60
    # run farm host market to use (ondemand, spot)
    run_instance_market: ondemand
    # If using spot instances, determine the interrupt behavior (terminate, stop, h
    # spot_interruption_behavior: terminate
    # If using spot instances, determine the max price
    spot_max_price: ondemand
    # default location of the simulation directory on the run farm host
    default_simulation_dir: /home/centos

    # run farm hosts to spawn: a mapping from a spot below (which is an EC2
    # instance type) to the number of instances of the given type that you
    # want in your pool(s).
    run_farm_hosts_to_spawn:
      - FL-1xlarge: 0
      - FL-2xlarge: 1
      - FL-3xlarge: 1
      - X1E-2xlarge: 0
      - X1E-3xlarge: 0
      - X1E-4xlarge: 0

  metasimulations:
    # metasimulations_enabled: false
    # use an verifier, use aws-ec2 as verifier-debug for hardware generation
    metasimulation_tool_simulator: verifier
    # @change passed to the simulator for all metasimulations
    # @change passed to the simulator ONLY FOR aws metasimulations
    metasimulation_only_verifier: "aws+integrate+aws+initmem"

  target_configs:
    # target: no_net_config
    no_net_no_net: 1
    link_latency: 6000
    autostop_latency: 10
    net_bmmemsize: 200
    output_format: 0

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

    # Advanced: Specify any extra @change you would like to provide when
    # booting the simulator in both FPGA-in and netx86 mode. This is
    # a string, with the contents formatted as if you were passing the @change
    # at command line, e.g. "next +>"
    @change_passthrough: ""

  tracing:
    enable: no

    # Trace output formats. Only enabled if 'enable' is set to 'yes' above
    # 0 = Human readable; 1 = binary (compressed raw data); 2 = flamegraph (stack
    # unrolling -> flame graph)
    output_format: 0

    # Trigger selector.
    # 0 = no trigger; 1 = cycle count trigger; 2 = program counter trigger; 3 =
    # instruction trigger
    selector: 1
    mask: 0
    unit: 0

  softwaretools:
    read_data: 0

  metadata:
    # metadata_name: linuxuniform.json
    # metadata_simulation: no
    # metadata_tag: null

```





Configuring a Build

config_build.yaml

```
■ Build-time build design / AGFI configuration for the FireSim
# See https://docs.firesim.com/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 behavior
    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

```
■ Build-time build recipe configuration for the FireSim Simulation Manager
# See https://docs.firesim.com/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`

```
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
```

Consists of:

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

```
WithNIC_DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSimHighPerfConfigTweaks_chipyard.QuadRocketConfig
```





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

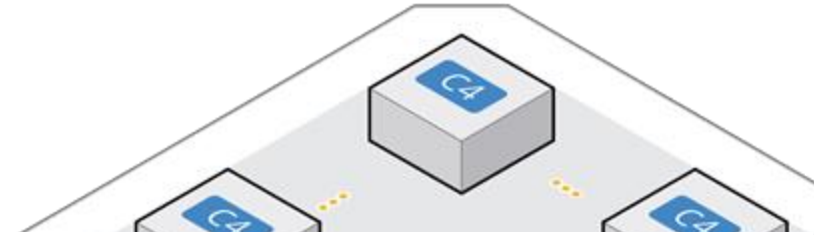


Running builds

- 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

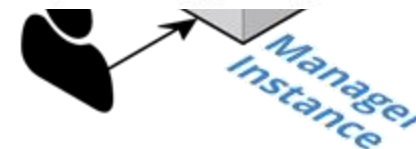
firesim_rocket_singlecore_no_nic_l2_lbp:

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>