



FireSim

Running a FireSim Simulation:
Password Cracking on a RISC-V SoC
with SHA-3 Accelerators and Linux

<https://firesim.com>



@firesimproject

ISCA Tutorial 2021

Albert Ou



Berkeley Architecture Research

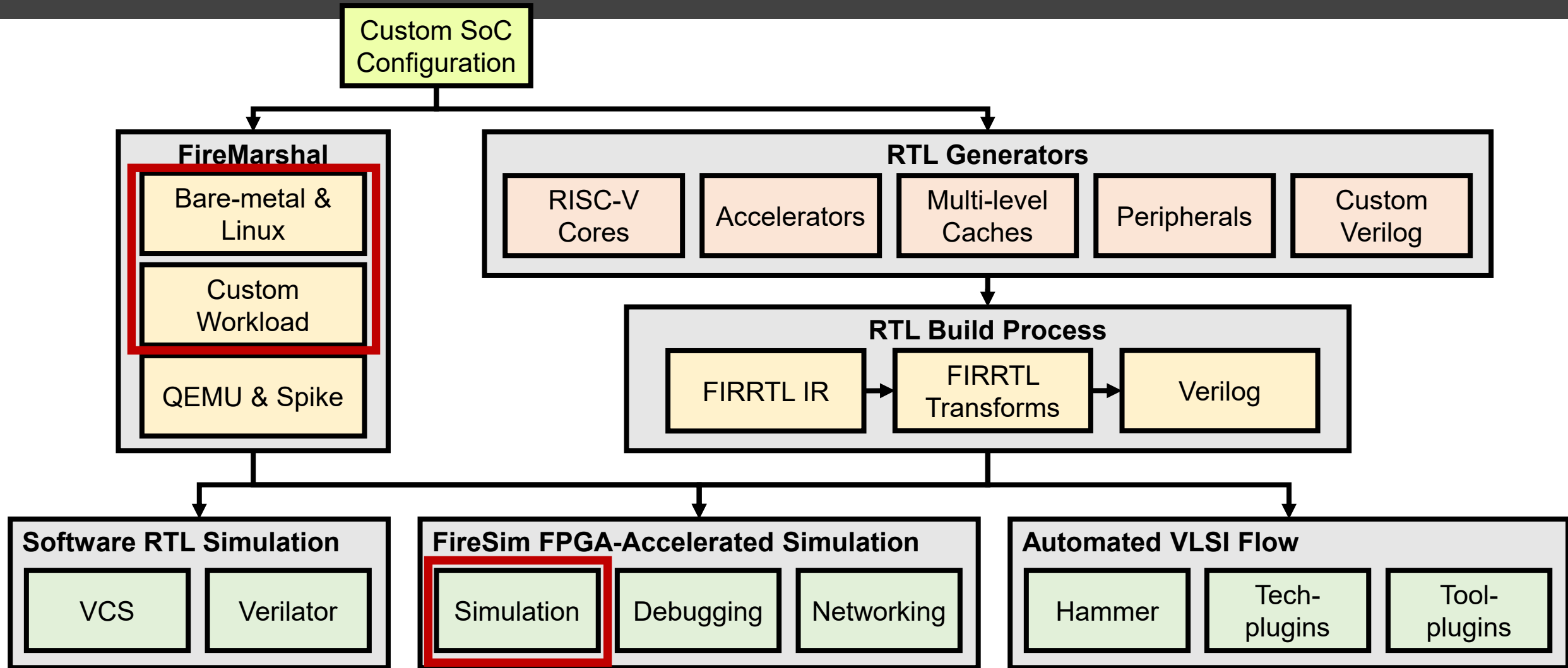


Agenda

- Configure and launch a simulation runfarm
- Boot Linux interactively on the target hardware
- Run custom automated workloads
- Demonstrate a complex Linux application (John the Ripper) that exercises an example SHA-3 accelerator



Tutorial Roadmap





Prerequisites

- Interactive shell commands are highlighted in blue blocks (prefixed with a “\$” prompt)
- Initialize the shell environment:
 - `FDIR` variable refers to the top FireSim directory within Chipyard

```
$ cd ~/chipyard
$ source ./env.sh

$ cd sims/firesim
$ export FDIR=$PWD
$ source ./sourceme-f1-manager.sh
```



Runtime Configuration

What to simulate and what infrastructure is required is controlled by
`$FDIR/deploy/config_runtime.ini`

- Target-level: Assemble a simulated system from components
 - FPGA images of SoC hardware designs
 - Network topology
 - Workload definition
- Host-level: Specify which EC2 instances to use

Refer to FireSim documentation for explanation of all parameters:
<https://docs.firesim/en/latest/Advanced-Usage/Manager/Manager-Configuration-Files.html>



config_runtime.ini

The `[runfarm]` section specifies the number, type, and other launch parameters of instances to be managed

```
[runfarm]
runfarmtag=mainrunfarm

f1_16xlarges=1
m4_16xlarges=0
f1_4xlarges=0
f1_2xlarges=0

runinstancemarket=ondemand
spotinterruptionbehavior=terminate
spotmaxprice=ondemand
```



config_runtime.ini

The `[targetconfig]` section specifies the high-level configuration of the system to simulate

```
[targetconfig]
topology=example_8config
no_net_num_nodes=2
linklatency=6405
switchinglatency=10
netbandwidth=200
profileinterval=-1

defaulthwconfig=firesim-rocket-quadcore-nic-12-11c4mb-ddr3
```

`defaulthwconfig` references an entry from `config_hwdb.ini`



config_runtime.ini

Configuration sections for various instrumentation facilities

- `[tracing]`: TracerV trace port capture
- `[autocounter]`: Out-of-band performance counter collection
- `[hostdebug]`: DRAM zeroing, synthesized assertions
- `[synthprint]`: Synthesized print statements



config_runtime.ini

The [workload] section specifies the software to be executed on the simulated system

```
[workload]
workloadname=linux-uniform.json
terminateoncompletion=no
suffixtag=
```

- Workload definitions located in `$FDIR/deploy/workloads/*.json`
- Some sample workloads are provided, including those to reproduce experiments from ISCA 2018 paper



Custom FireSim Workloads

- *Workload*: Specification for a series of jobs (software configurations) assigned to run on individual simulated nodes
- Two types of workloads:
 - Uniform**: Homogenous job run by all nodes in a simulated cluster
 - Non-uniform**: Each node is assigned a different job
 - Client/server configurations
 - Benchmark suites (**SPEC CPU2017**)
- Can be manually written or generated by FireMarshal



Workload Definitions

`linux-uniform`: Default workload to boot an interactive buildroot-based GNU/Linux distro on every node

```
{
  "benchmark_name" : "linux-uniform",
  "common_bootbinary" : "br-base-bin",
  "common_rootfs" : "br-base.img",
  "common_outputs" : ["/etc/os-release"],
  "common_simulation_outputs" : ["uartlog", "memory_stats.csv"]
}
```

`$FDIR/deploy/workloads/linux-uniform/br-base{-bin,.img}`
are symlinks to the FireMarshal-generated images



Building linux-uniform

`linux-uniform` corresponds to `br-base` FireMarshal workload

```
$ cd ~/chipyard/software/firemarshal  
$ marshal -v build boards/firechip/base-workloads/br-base.json
```



Single-Node Simulation

Edit `config_runtime.ini` to match the following settings:

```
[runfarm]
f1_16xlarges=0
f1_2xlarges=1
```

- Use a smaller f1.2xlarge instance (1 FPGA)
- Simulate one non-networked node without a switch model
- Deploy the pre-built quad-core Rocket design without a NIC

```
[targetconfig]
topology=no_net_config
no_net_num_nodes=1
```

```
default_hwconfig=firesim-rocket-quadcore-no-nic-l2-11c4mb-ddr3
```



Launching Simulation Instances

```
$ firesim launchrunfarm
```

```
FireSim Manager. Docs: http://docs.firesim.com
```

```
Running: launchrunfarm
```

```
Waiting for instance boots: 0 f1.16xlarges
```

```
Waiting for instance boots: 0 f1.4xlarges
```

```
Waiting for instance boots: 0 m4.16xlarges
```

```
Waiting for instance boots: 1 f1.2xlarges
```

```
i-053d65f4b514170be booted!
```

```
The full log of this run is:
```

```
/home/centos/chipyard/sims/firesim/deploy/logs/2021-06-18--00-32-29-launchrunfarm-VAEU6Y2339M00YH1.log
```



Deploying Simulation Infrastructure

```
$ firesim infrasetup
```

This deploys various software prerequisites:

- Builds host-side simulation drivers for the specific build triplet
- Builds the switch model executable (if enabled)
- Collects information about simulation instances and transfers files
- Builds and loads the XDMA kernel driver
- Loads the NBD kernel driver for mounting qcow2 disk images
- Programs the FPGAs with the desired AGFIs



Deploying Simulation Infrastructure

```
$ firesim infrasetup
```

```
FireSim Manager. Docs: http://docs.firesim
Running: infrasetup

Building FPGA software driver for FireSim-DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSimTestChipConfigTweaks_chipyard.QuadRocketConfig-
WithAutoILA_F90MHz_BaseFlConfig
[192.168.3.73] Executing task 'instance_liveness'
[192.168.3.73] Checking if host instance is up...
[192.168.3.73] Executing task 'infrasetup_node_wrapper'
[192.168.3.73] Copying FPGA simulation infrastructure for slot: 0.
[192.168.3.73] Installing AWS FPGA SDK on remote nodes. Upstream hash: 6c707ab4a26c2766b916dad9d40727266fa0e4ef
[192.168.3.73] Unloading XDMA/EDMA/XOCL Driver Kernel Module.
[192.168.3.73] Copying AWS FPGA XDMA driver to remote node.
[192.168.3.73] Unloading XDMA/EDMA/XOCL Driver Kernel Module.
[192.168.3.73] Loading XDMA Driver Kernel Module.
[192.168.3.73] Setting up remote node for qcow2 disk images.
[192.168.3.73] Unloading NBD Kernel Module.
[192.168.3.73] Disconnecting all NBDs.
[192.168.3.73] Loading NBD Kernel Module.
[192.168.3.73] Clearing FPGA Slot 0.
[192.168.3.73] Checking for Cleared FPGA Slot 0.
[192.168.3.73] Flashing FPGA Slot: 0 with agfi: agfi-0a6cbb217f7cc16d6.
[192.168.3.73] Checking for Flashed FPGA Slot: 0 with agfi: agfi-0a6cbb217f7cc16d6.
[192.168.3.73] Unloading XDMA/EDMA/XOCL Driver Kernel Module.
[192.168.3.73] Loading XDMA Driver Kernel Module.
[192.168.3.73] Starting Vivado hw_server.
[192.168.3.73] Starting Vivado virtual JTAG.
The full log of this run is:
/home/centos/chipyard/sims/firesim/deploy/logs/2021-06-18--00-44-47-infrasetup-3DIBA22C96MNT5GN.log
```





Running the Simulation

```
$ firesim runworkload
```

```
FireSim Manager. Docs: http://docs.firesim.im  
Running: runworkload
```

```
Creating the directory: /home/centos/chipyard/sims/firesim/deploy/results-  
workload/2021-06-18--01-02-45-linux-uniform/  
[192.168.3.73] Executing task 'instance_liveness'  
[192.168.3.73] Checking if host instance is up...  
[192.168.3.73] Executing task 'boot_switch_wrapper'  
[192.168.3.73] Executing task 'boot_simulation_wrapper'  
[192.168.3.73] Starting FPGA simulation for slot: 0.  
[192.168.3.73] Executing task 'monitor_jobs_wrapper'
```



Monitoring the Simulation

You should see a live status report that refreshes periodically:

```
FireSim Simulation Status @ 2021-06-18 01:03:54.145398
-----
This workload's output is located in:
/home/centos/chipyard/sims/firesim/deploy/results-workload/2021-06-18--01-02-45-
linux-uniform/
This run's log is located in:
/home/centos/chipyard/sims/firesim/deploy/logs/2021-06-18--01-02-45-runworkload-
8Y0DIA0UA7RIEBTL.log
This status will update every 10s.
-----
Instances
-----
Instance IP:   192.168.3.73 | Terminated: False
-----
Simulated Switches
-----
Simulated Nodes/Jobs
-----
Instance IP:   192.168.3.73 | Job: linux-uniform0 | Sim running: True
-----
Summary
-----
1/1 instances are still running.
1/1 simulations are still running.
-----
```





Interacting with the Simulation

Look for the run instance's IP address in the status:

```
FireSim Simulation Status @ 2021-06-18 01:03:54.145398
-----
This workload's output is located in:
/home/centos/chipyard/sims/firesim/deploy/results-workload/2021-06-18--01-02-45-
linux-uniform/
This run's log is located in:
/home/centos/chipyard/sims/firesim/deploy/logs/2021-06-18--01-02-45-runworkload-
8Y0DIA0UA7RIEBTL.log
This status will update every 10s.
-----
Instances
-----
Instance IP: 192.168.3.73 | Terminated: False
-----
Simulated Switches
-----
Simulated Nodes/Jobs
-----
Instance IP: 192.168.3.73 | Job: linux-uniform0 | Sim running: True
-----
Summary
-----
1/1 instances are still running.
1/1 simulations are still running.
-----
```





Logging Into the Simulated System

- Once Linux boots, the login prompt should appear over the console
- Log in as `root` with password `firesim` (password does not echo)

```
[    0.085714] EXT4-fs (iceblk): re-mounted. Opts: (null)
Starting syslogd: OK
Starting klogd: OK
Starting mdev... done.
Starting dropbear sshd: OK

Welcome to Buildroot
buildroot login: root
Password:
#
```



Logging Into the Simulated System

- Feel free to experiment with shell commands

```
# uname -a  
# cat /proc/cpuinfo  
# free -m  
# vim
```

- When done, cleanly shut down the system

```
# poweroff
```

- This will then end the simulation automatically



John the Ripper

- Open-source password cracking software
- Our customized version adds support for two more hash formats:
 - **Raw-SHA3-256**: pure software implementation using generic Keccak code
 - **Raw-SHA3-256-rocc**: RoCC accelerator offload
- <https://github.com/ucb-bar/JohnTheRipper/tree/riscv>
 - `JohnTheRipper/src/sha3_256_rocc_fmt_plug.c`
 - The `crypt_all()` function performs the actual hashing
- Minor Linux kernel patches to facilitate accelerator context switching



Defining a New Hardware Configuration

John the Ripper requires a custom target configuration that instantiates the SHA-3 accelerator

- Add to `config_build_recipes.ini`:

```
[firesim-rocket-singlecore-sha3-no-nic-12-11c4mb-ddr3]
DESIGN=FireSim
TARGET_CONFIG=DDR3FRFCFSLLC4MB_WithDefaultFireSimBridges_WithFireSimConfigT
weaks_ chipyard.Sha3RocketConfig ←
PLATFORM_CONFIG=WithAutoILA_F110MHz_BaseF1Config
instancetype=z1d.2xlarge
deploytriplet=None
```

Same SHA3-accelerated SoC design generated during the morning Chipyard session



Building a New Hardware Configuration

- Set in `config_build.ini`:

```
[builds]
firesim-rocket-singlecore-sha3-no-nic-12-11c4mb-ddr3
# Comment out all other entries
```

- Build custom AGFI:

```
$ firesim buildagfi
```

- Update `config_hwdb.ini`:

```
$ cd $FDIR/deploy
$ cat built-hwdb-entries/firesim-*-sha3-* >> config_hwdb.ini
```



Configuring a New Workload

- Generate the FireSim workload definition for “sha3-linux-jtr-test”:

```
$ cd ~/chipyard/generators/sha3/software  
$ marshal -v build marshal-configs/sha3-linux-jtr-test.json  
$ marshal install marshal-configs/sha3-linux-jtr-test.json
```

- Update `config_runtime.ini` accordingly:

```
defaulthwconfig=firesim-singlecore-sha3-no-nic-12-11c4mb-ddr3  
workloadname=sha3-linux-jtr-test.json
```

```
$ firesim infrasetup  
$ firesim runworkload
```



Basic Benchmarking

- The workload first runs John the Ripper's low-level self-tests and benchmarks to measure raw hash performance
 - Passwords constitute a less optimal input for the accelerator
 - Many unrelated messages much shorter than the block size (1088 bits)
- “Crypts per second” (C/s) metric
 - *Real*: elapsed real time
 - *Virtual*: total CPU time

```
Benchmarking: Raw-SHA3-256 [SHA3 256 32/64]... DONE
Raw:      66423 c/s real, 66756 c/s virtual
```

```
Benchmarking: Raw-SHA3-256-rocc [SHA3 256 32/64]... DONE
Raw:      3037K c/s real, 3037K c/s virtual
```



Password Cracking

- In the second half of the workload, the SHA-3 accelerator is used to attack sample hashes from the default wordlist
- `john` is given input files of one hash per line, unsalted for simplicity:

```
hash_0:be87f99a67e48ec4ec9f05b565f6ca531e24b9c71a62cfd3a58f54ebc60115ea
hash_1:f706280cdf972ed4af636d540e7d2ea2ff3e9f91e63bc389b2aa0fa288c486a9
hash_2:2cd81e6887b1618af765e2bc127f68b563e6a1b4abd397331b759f878eb8515e
hash_3:9cdc6b9ff3d0d0a90cb8670fb972debc08947697c6b63903458abbaaa0fe93c9
```

- The companion “`sha3-linux-jtr-crack`” workload includes a more challenging scenario that tests the incremental (brute-force) mode



Capturing Results

- Once the workload terminates automatically, the results are copied to the manager instance:

```
FireSim Simulation Exited Successfully. See results in:  
/home/centos/chipyard/sims/firesim/deploy/results-workload/2021-06-18--01-09-42-  
sha3-linux-jtr-test/
```

- The exact directory path will contain a different timestamp
- Console output recorded in `sha3-linux-jtr-test0/uartlog`
 - As well as other output files named in the workload definition



Password Cracking

- “sha3-linux-jtr-crack” includes two different tests
 1. A short test based on passwords present in the default wordlist
 2. A long test that resorts to incremental (brute-force) mode
- Repeat the same procedure as “sha3-linux-jtr-test”
 - Use the same AGFI: `firesim-singlecore-sha3-no-nic-12-11c4mb-ddr3`
 - Generate the workload definition with `marshal`
 - Edit `config_runtime.ini` to reference this workload
- Start the simulation and attach the UART console to watch progress



Killing a Simulation

- Brute-forcing the last hash takes a few minutes of simulated time
- Too long to complete at this tutorial
- To stop a simulation in the middle:

```
$ firesim kill
```

```
FireSim Manager. Docs: http://docs.firesim  
Running: kill
```

```
[192.168.3.73] Executing task 'kill_switch_wrapper'  
[192.168.3.73] Executing task 'kill_simulation_wrapper'  
[192.168.3.73] Killing FPGA simulation for slot: 0.  
[192.168.3.73] Disconnecting all NBDs.  
[192.168.3.73] Executing task 'screens'  
Confirming exit...
```



Terminating Simulation Instances

```
$ firesim terminatorunfarm
```

```
FireSim Manager. Docs: http://docs.firesim.com  
Running: terminatorunfarm
```

```
IMPORTANT!: This will terminate the following instances:
```

```
f1.16xlarges
```

```
[]
```

```
f1.4xlarges
```

```
[]
```

```
m4.16xlarges
```

```
[]
```

```
f1.2xlarges
```

```
['i-053d65f4b514170be']
```

```
Type yes, then press enter, to continue. Otherwise, the operation will be cancelled.
```

```
yes
```

```
Instances terminated. Please confirm in your AWS Management Console.
```

Type **yes** at the prompt to confirm



Workflow Summary

1. Build an AGFI (“Building Hardware Designs in FireSim” session)
2. Build target software (“Building Software Workloads with FireMarshal” session)
3. Customize `config_runtime.ini`
4. `firesim launchrunfarm`
5. `firesim infrasetup`
6. `firesim runworkload`
7. `firesim terminatorunfarm`