# Simulation Environment Documentation

This document explains the simulation environment for the project, including directory structure, configuration systems, automation scripts, run tagging, and results handling.

## 1. Overview

The environment supports:

- Running SimpleScalar benchmarks
- Running ChampSim traces
- Configurable cache parameters
- Versioned experiments
- Automatic results/csv generation
- Extensibility for new cache designs

## 2. Directory Structure

```
Project/
    env.sh
    scripts/
    config/
    results/
    traces/
    simplesim-3.0/
    ChampSim/
    docs/
```

All experiments are executed through shells scripts that source `env.sh` for a consistent runtime environment

### Key components:

- `env.sh` - Sources environment
- `scripts/` - SS and ChampSim automation
- `config/` - User-editable experiment settings
- `results/` - Tagged output directories
- `traces/` - Benchmarks and inputs
- `docs/` - Documentation

## Run Tagging

The environment allows for run tagging to track simulations and runs.

Usage:

```
export RUN_TAG=baseline
source env.sh
```

**All autojmation scripts assume `env.sh` has already been sourced and rely on `RUN_TAG` to determine output directories.**

This will automatically set

```
SS_RESULTS = results/ss/baseline
CHAMPSIM_RESULTS = results/champsim/baseline
```

Each experiment or run can be assigned its own tag to organize results

## 4. Configuration Files

Configuration files are located in `config/`

### 4.1: SimpleScalar Cache Geometry - `ss_cache.sh`

```
export SS_IL1_CONFIG="il1:32768:64:1:l"
export SS_DL1_CONFIG="dl1:32768:64:1:l"
export SS_UL2_CONFIG="ul2:262144:64:4:l"
```

### 4.2: SimpleScalar Benchmarks - `ss_benchmarks.txt`

```
test-fmath traces/test-fmath
test-llong traces/test-llong
test-lswlr traces/test-lswlr
test-math  traces/test-math
```

Each line specifies a benchmark name followed by the executable path relative to the project root.

### 4.3: ChampSim Runtime Settings - `champsim_run.sh`

```
export CS_BINARY_NAME="champsim"
export CS_WARMUP_INS=1000000
export CS_SIM_INS=10000000
```

### 4.4: ChampSim Trace List - `champsim_traces.txt`

```

```

```
600.perlbench_s-210B.champsimtrace.xz
602.gcc_s-216B.champsimtrace.xz
429.mcf-184B.champsimtrace.xz
```

To change a workload, only these files need to be edited. The scripts themselves do not need to be edited.

## 5. Automation Scripts

### SimpleScalar

- `run_ss_cache.sh` - run a single benchmark
- `run_ss_sweep.sh` – runs all SimpleScalar benchmarks listed in `ss_benchmarks.txt` using the cache configuration and experiment mode defined by the current environment.

**All sweep scripts assume the environment has been initialized via `source env.sh`

Outputs:

- per-benchmark logs
- `ss_baseline.csv` summarizing miss rates and CPI

### ChampSim

- `run_champsim.sh` - run a single trace
- `run_champsim_sweep.sh` - run all configured traces

Outputs:

- per-trace logs
- `champsim_baseline.csv` with L1I, L1D, LLC miss rates

## 6. Results Directories

All results are stored by a tag: Example:

```
results/
  ss/
    baseline/
    victim/
  champsim/
    baseline/
    victim/
```

Each subdirectory corresponds to a unique `RUN_TAG` and may contain multiple CSV files, one per cache mechanism or experiment mode.

## 7. Using the Environment

Example:

```
export RUN_TAG=example_tag
source env.sh
ss_sweep
cs_sweep
```

The `ss_sweep` command is typically an alias or wrapper script that invokes `run_ss_sweep.sh`

## Notes on Reproducibility

All experiments are fully reproducible by recording:

- The `RUN_TAG`
- Cache configuration files under `config/`
- The automation script used to launch the experiment

No simulator source code changes are required to reproduce results once the environment is configured.