Project 2025

This document contains the instructions and commands to setup Homework 5 directory. This Homework make use of cmake build system to build and run the common command used eary in the course.

Overview

- Decompress
- Start Designing
- Synthesis
- Submission
- Appendix

Decompress

Once you have placed `projectFall2025.tar.gz` at desired directory. Launch a terminal at that directory and use the following command to tar.

```
tar xzf projectFall2025.tar.gz
```

You should find the unzipped project folder `projectFall2025/`

Start Designing

Setup script

`projectFall2025/setup.sh` is to load Modelsim and Synopsys. You can use differnt version if you want, but this verison is the one that will be used ot test you design.

To source the script:

```
cd projectFall2025
source setup.sh
```

This script also enables you to Tab complete `make` commands

Homework description

The document is located in `projectFall2025/Project_specification/`

Where to put your design

A Verilog file `projectFall2025/srcs/rtl/dut.sv` is provided with all the ports already connected to the test fixture. The dut.sv is a stub to put your code in. The detailed rundown of the directory structure can be found in the **Appendix** of this document.

How to setup build environment

To setup you build environment

```
cd projectFall2025
source setup.sh
mkdir build
cd build
cmake .. --preset run
```

You will see a message showing the default Unity ID.

```
DC_CLOCK_PER="4.0"
TEST="-1"
UNITY_ID="jdoe12"
CLASS="ECE564"
```

Update the variables in the CMakePresets.json file using your preferred text editor. Edit line 11, 12 to set your Unity ID and CLASS ID.

```
"UNITY_ID": "jdoe12", -> "UNITY_ID": "your_id",
"CLASS": "ECE564", -> "CLASS": "ECE464",
```

How to compile your design

This project provides three main simulation targets:

- **dut** — Build and run **your implementation**
  - Synthesizable RTL intended for final submission
  - Must meet interface timing and support pipelining
  - Use this target to verify functionality, latency, and protocol compliance
- **golden** — Run the **reference SystemVerilog model**
  - **Not pipelined** and **not synthesizable**

  - Behavioral-only model used as a correctness oracle

- **sram** — Run the **memory test suite**
  - Demonstrates expected behavior for back-to-back commands
  - Helps visualize correct response sequences and edge-case handling

---

How to run simulation targets

Normal and Debug targets

There are two input types normal and debug. Normal is the inputs that you will be graded on, debug is used to help you design your module.

Configure for Normal input:1024x1024

```
cd projectFall2025/build
cmake .. --preset run
```

Configure for Debug input:32x32; it's just for convenience and not a design requirement.

```
cd projectFall2025/build
cmake .. --preset debug
```

Run with Modelsim UI:

```
make vsim-dut
```

This buids `projectFall2025/srcs/rtl/dut.sv` and launches the testbench in ModelSim.

Build your implementation (dut)

This command can be executed inside ModelSim.

```
make vlog-dut
```

The following command when executed in ModelSim, will reload, restart, and run all the test cases after you make changes to the dut file.

```
make vlog-dut; restart -f; run -all
```

Simulation

The simulation log file are here:

`projectFall2025/build/sim/logs/result.log`

The simulation output files are here

`projectFall2025/build/sim/outputs/output[0/1/2/3].dat`

There is a do file that you can update for your design `projectFall2025/do/waves.do`

In Modelsim: This will add the waves in you design

```
do ../do/waves.do
```

Run golden model with Modelsim UI

```
make vsim-golden
```

All other commands are the same as the your implementation (dut)

Run memory behavior tests with Modelsim UI

```
make vsim-sram
```

This is a short example demonstrating the memory behavior, in case you are not sure how to interact with the sram module.

Configuring for individual tests:

The following command will configure `vlog-dut` to run only one of the test cases, `<test_case>` can be 0, 1, 2, 3, 4 or 5. In case your design only fails part of the test cases, this can be used to speed up the run.

```
cmake .. --preset test<test_case>
make vsim-dut
```

An example configuring `vlog-dut` to run only test case 2 is shown as follows:

```
cmake .. --preset test2
make vsim-dut
```

Configuring for all tests:

Setting `run` will configure `vlog-dut` to run all test cases.

```
cmake .. --preset run
make vsim-dut
```

Configuring for individual debug:

The following command will configure `vlog-dut` to run only one of the debug cases, `<test_case>` can be 0, 1, 2, or 3. In case your design only fails part of the test cases, this can be used to speed up the run.

```
cmake .. --preset debug<test_case>
make vsim-dut
```

An example configuring `vlog-dut` to run only test case 2 is shown as follows:

```
cmake .. --preset debug2
make vsim-dut
```

Configuring for all debugs:

Setting `debug` will configure `vlog-dut` to run all debug cases.

```
cmake .. --preset run
make vsim-dut
```

Evaluation Testing

To evaluate your design headless/no-gui, change directory to `projectFall2025/build/`

```
cmake .. --preset run
make vsim-dut-c # Runs the test in headless mode
```

This will produce a log file that will highlight the results of your design. This should only be ran as a final step before Synthesis

Simulation log file is located here `projectFall2025/build/sim/logs/results.log`

Synthesis

Once you have a functional design, you can synthesize it in `projectFall2025/build/`

Synthesis Command

The following command will synthesize your design with a default clock period of 10 ns

```
make synth
```

Clock Period

To run synthesis with a different clock period

```
cmake .. -DDC_CLOCK_PER=YYY.XX
make synth
```

For example, the following command will set the target clock period to 4 ns.

```
cmake .. -DDC_CLOCK_PER=4.0
make synth
```

Synthesis Reports

You can find your timing report and area report in `projectFall2025/build/synth/reports/`

Submission

Project Report

Place your report file in `projectFall2025/Project_report/`. Replace the report.pdf with your report

Edit line `13` in `projectFall2025/CMakePresets.json` to reflect the CLK_PER that you used to synthize your design.

```
"DC_CLOCK_PER": "10.0" -> "DC_CLOCK_PER": "4.0"
```

Compress for submission

To generate the tar.gz file for submission

change directory to `projectFall2025/build` and use the following command

```
cmake .. --preset run
make compress
```

Make sure the message reflects the your unity id and new clk and class type.

```
DC_CLOCK_PER="10.0" ->  DC_CLOCK_PER="YYY.XXX"
TEST="-1"
UNITY_ID="jdoe12" -> UNITY_ID="your_id"
"CLASS": "ECE564", -> "CLASS": "ECE464",
```

You will find the generated tar.gz file in `projectFall2025/build/submission.your_id.tar.gz`

Check before you submit

Please check your .tar.gz file and make sure all the files are present for submission

The following files are need to be able to run your desing:

```
projectFall2025/srcs/rtl/dut.sv
projectFall2025/CMakePresets.json
```

Check to make sure you report is in the following dir:

```
projectFall2025/Project_report/*.pdf
```

Your design simulation will be evaluated by, 1st: downloading the tar, 2nd: running these commands:

```
cd projectFall2025
cp ./path/to/tar/file/submission.jdoe12.tar.gz ./
tar xzf submission.your_id.tar.gz // This command will update the files for your design
mkdir build
cd build
cmake .. --preset run
make vsim-dut-c synth
```

It's recommended to download a fresh copy of the projectFall2025 directory and run the command above.

Submit your files

Upload the generated `projectFall2025/build/submission.your_id.tar.gz` file to Moodle page

Appendix

Directory Rundown

You will find the following directories in `projectFall2025/`

- `projectFall2025/`
  - Contains the `CMakeLists.txt` used to compile and simulate the design
- `projectFall2025/inputs/input[0/1/2/3/4/5].dat`
  - Contains the .dat files for the memory content for DRAM used in Project
- `projectFall2025/outputs/output[0/1/2/3/4/5].dat`
  - Contains the .dat files for the output DRAM used in Project
- `projectFall2025/srcs/rtl/`
  - All .v and .sv files will be compiled when executing `make vlog-dut` in `projectFall2025/build/`
  - A template `dut.sv` that interfaces with the test fixture is provided
- `projectFall2025/srcs/tb/golden_dut.sv`
  - Contains the reference behavior model for the Project
  - Use this commands to build the golden model `make vlog-golden` in `projectFall2025/build/`
- `projectFall2025/Project_report/`
  - Replace projectFall2025/Project_report/report.pdf with your Project report here before running `make compress` command
- `projectFall2025/Project_specification/`
  - Contains the Project specification document
- `projectFall2025/synthesis/`
  - The directory where the tcl scripts used to synthesize your design
  - Only the CompileAnalyze.tcl will be include in the submission.<your_unity_id>.tar.gz
  - Synthesis reports will be exported to `projectFall2025/build/synth/reports/`
  - Synthesis logs will be exported to `projectFall2025/build/logs/reports/`
  - Synthesized netlist will be generated to `projectFall2025/build/synth/gl/`
- `projectFall2025/srcs/tb/`
  - Contains the test fixture of the Project