

## **SimpleScalar simulation of computer architecture**

**Work in teams of three or four.**

**You decide your own team members and post to the wall your team members.**

**You can work solo if you want.**

**No questions are allowed since it is considered mini project. You can assume any missing information that will help you to achieve this task. For graduate students, you have the freedom to pick the problem that you want for this assignment.**

**Submit a pdf file of the procedure of installing and setting up the software, the code, print screen of the outputs, and both the problem you want to investigate and your conclusion.**

SimpleScalar tool is a system software infrastructure used to build modeling applications for program performance analysis, detailed micro-architectural modeling, and hardware-software co-verification.

For software developers, SimpleScalar provides detailed performance information about program running on a variety of target platforms with a single host simulation platform.

It provides sample simulators ranging from a fast functional simulator to a detailed, dynamically scheduled processor model that supports non-blocking caches, speculative execution, and state-of-the-art branch prediction.

For assignment 4, students need to work in teams. Each team can be  $n$  students where  $n$  will be determined. The goal is to investigate SimpleScalar simulation tool, understand it, utilize it to simulate any concept covered in our course, and finally report your findings, results, and documentation in your final report.

Final report will be submitted by one member in behalf of the group.

An example of a part of the documentation is given below in Appendix A.

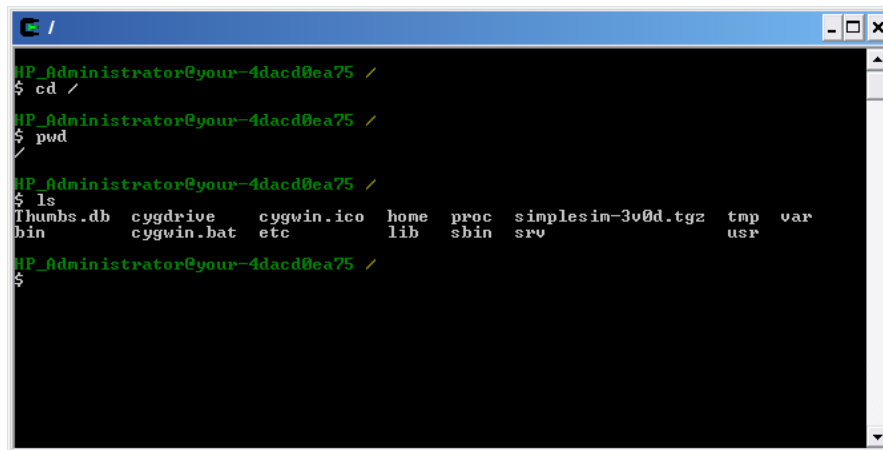
A sample task that you can implement in SimpleScalar is given in Appendix B.

You do not have to follow any information from the Appendix. Follow your own research and installation and propose your own application.

# Appendix A:

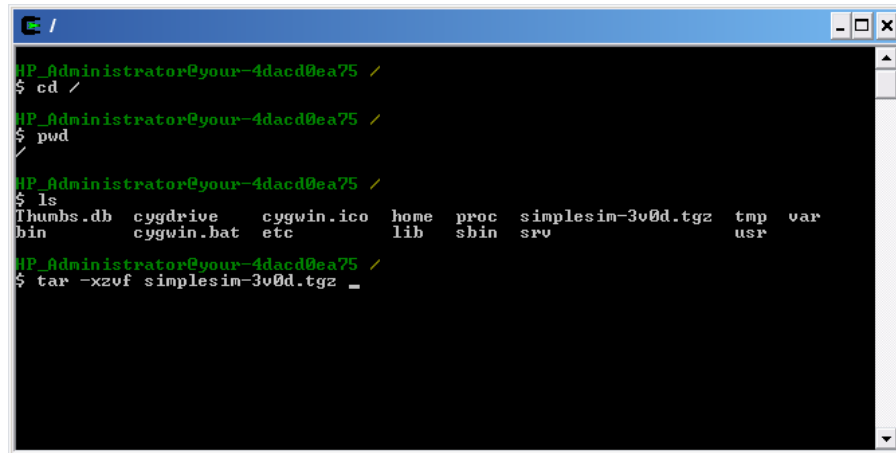
## SimpleScalar Installation --- For Cygwin on Windows/PC platform

1. Get the SimpleScalar package from <http://www.simplescalar.com>
2. Go to Tools in the Downloads section to the left and download simplesim-3v0d.tgz.
3. Download the package directly into Cygwin's root directory (c:\cygwin).
4. Open the Cygwin and make sure that you are at Cygwin root directory as shown below. Just type `cd /` and press enter to make sure you are at Cygwin root directory.
5. Type `ls` and press enter to see if the downloaded file is there. As shown below



```
HP_administrator@your-4dacd0ea75 /
$ cd /
HP_administrator@your-4dacd0ea75 /
$ pwd
/
HP_administrator@your-4dacd0ea75 /
$ ls
Thumbs.db  cygdrive  cygwin.ico  home  proc  simplesim-3v0d.tgz  tmp  var
bin        cygwin.bat  etc        lib   /sbin  srv      usr
```

6. Type `tar -xzf simplesim-3v0d.tgz` and press enter as shown below to untar and unzip the package to install SimpleScalar.



```
HP_Administrator@your-4dacd0ea75 /
$ cd /
HP_Administrator@your-4dacd0ea75 /
$ pwd
/
HP_Administrator@your-4dacd0ea75 /
$ ls
Thumbs.db  cygdrive  cygwin.ico  home  proc  simplesim-3v0d.tgz  tmp  var
bin        cygwin.bat  etc        lib    shin  srv          usr
HP_Administrator@your-4dacd0ea75 /
$ tar -xvzf simplesim-3v0d.tgz _
```

.....additional steps are removed.....This step is to verify that the simulators built OK  
..... The SimpleScalar installation is complete.

## Appendix B:

An application or idea that you can implement in SimpleScalar is:  
A Three-level Cache Simulation using SimpleScalar with Least Frequency Used Replacement Policy

### Objective

- Implement a 3-level cache (L1, L2, and L3) using SimpleScalar. L1 cache consists of separate I and D caches, both L2 and L3 are unified caches
- Support replacement policy which is LFU
- Perform experiments to evaluate the cache design
- Use at least 3 benchmarks to collect the cache hit/miss information for two different configurations
- Analyze the performance

### Implementation

The implementation of an L3 cache for SimpleScalar requires few changes to the simulator. The changes will be confined to the *sim-cache.c* file and the configuration files. The configuration files will be modified to include the extra parameter to include the L3 option. The major modification to the file will be to change the cache hierarchy. Rather than having the L2 cache go to memory on a miss, the team will insert a check for an L3 cache. If the L3 exists, it should go there to look for the data rather than directly to memory.

