|  |  |  |  |
| --- | --- | --- | --- |
| Assignment | | | |
| Project name | SA2 | | |
| Document ref |  | | |
| Version |  | | |
| Release date |  | | |
| Author | Wu Fei | | |
| Classification |  | | |
| Distribution List |  | | |
| Approved by | Name | Signature | Date |
|  |  |  |  |



**Ming Hsieh Department of Electrical Engineering**

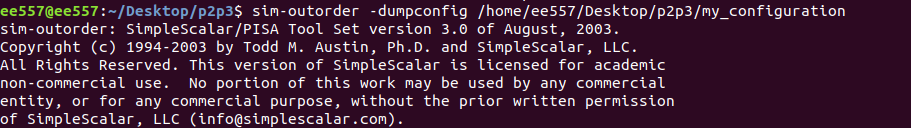
**University of Southern California, Los Angeles, CA 90089**

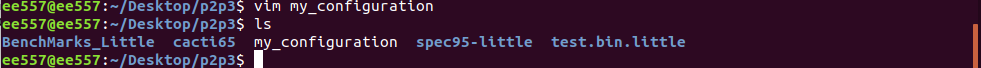
**Fall 2022**

**EE 557**

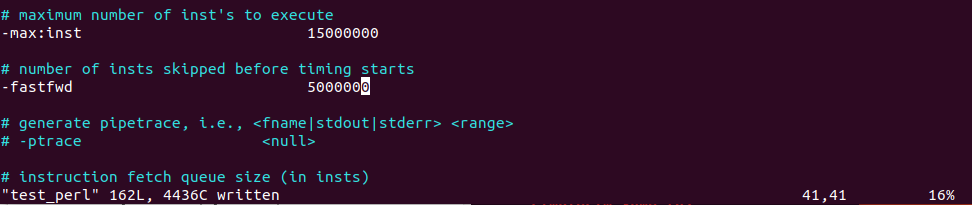
# Processing

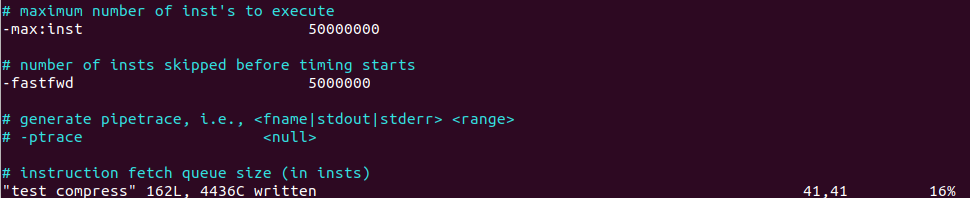
1. run SimpleScalar from your personal directory and generate a configuration file:





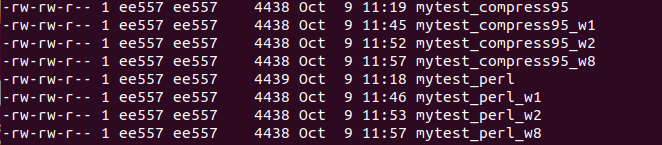
1. Adding constrain for both simulation files:





1. Using Real Estimator to get the entry size and some other parameter we need
2. Calculate the access time with the parameter we get in 4
3. Get the latency needed by the configuration file we generator in 1
4. Complete the configuration file and using SimpleScalar Simulator to calculate the processor performance
5. Change the machine widths as 1/1/1 2/2/2 3/3/3 4/4/4

Redo 3 to get the corresponding parameters and generate some more configuration files for different machine widths:



1. Get all the performance and resource usage with the SimpleScalar of the four machine widths and reach the conclusion in the following report.

# Points hit

For 2.2 I have package all the result from the cacti to files named as “rst\_” with -->;

**All the files mentioned in the report are submitted within the zip file p2p3.**

## Setting configuration file parameters

I have created a baseline configuration file named as my\_configuration under the p2p3 folder. And the parameters inside have been specified as the section III.d as the requirement.

**The configuration files are in a text file named mytest\_perl and mytest\_compress95.**

## L1-I, L1-D and L2 caches access times from Cacti, areas from Cacti and areas from Real Estimator in a table: 0.5 pt.

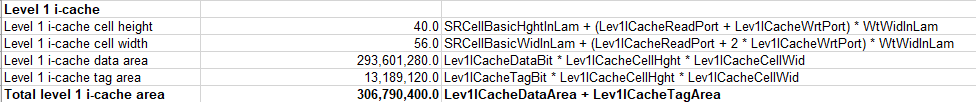
L1-I, L1-D and L2 caches access times from Cacti

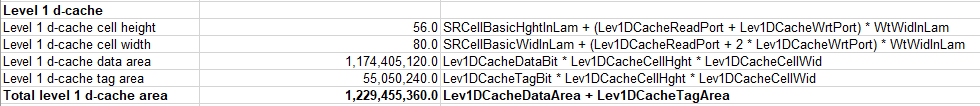
|  |  |  |  |
| --- | --- | --- | --- |
|  | L1-I | L1\_D | L2 |
| caches access times from Cacti (ns) | 1.20055 | 1.67861 | 2.84891 |

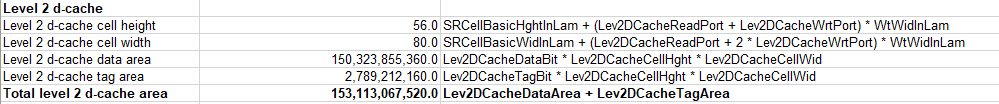
areas from Cacti

|  |  |  |  |
| --- | --- | --- | --- |
|  | L1-I | L1\_D | L2 |
| Cache height x width (mm) | 0.257445 x 0.107855 | 0.492636 x 0.25059 | 3.1878 x 2.31406 |
| Data array: Area (mm2) | 0.0184298 | 0.0700688 | 5.63072 |
| Tag array: Area (mm2) | 0.00305056 | 0.0128898 | 0.383327 |

areas from Real Estimator





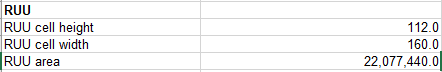


## RUU access time from Cacti, area from Cacti and area from Real Estimator for the machine width, 1, 2, 4 and 8 in a table: 0.5 pt.

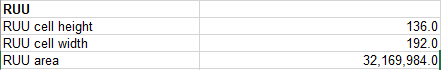
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1/1/1/1 | 2/2/2/2 | 4/4/4/4 | 8/8/8/8 |
| caches access times from Cacti (ns) | 1.01899 | 1.10035 | 1.25804 | 1.50735 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1/1/1/1 | 2/2/2/2 | 4/4/4/4 | 8/8/8/8 |
| Cache height x width (mm) | 0.859818 x 0.0968127 | 1.09515 x 0.11817 | 1.56671 x 0.1609 | 2.50957 x 0.246349 |
| Data array: Area (mm2) | 0.0832413 | 0.129419 | 0.252083 | 0.618231 |

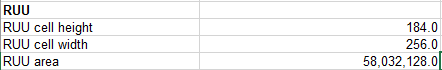
1/1/1/1



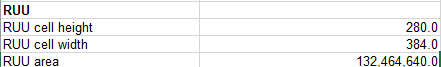
2/2/2/2



4/4/4/4



8/8/8/8



## Description of how the clock cycle time, L1-I latency, L1-D latency and L2 latency were obtained for the machine width 4: 1pt

The clock cycle time is RUU access time. Clock cycle time and latency were obtained as following:

First using Real Estate Estimator, insert the processor parameters provided and default in the first sheet and the tool will calculate and get the entry size for RUU and caches.

Then using Cacti, with the data calculated above, such as cache size, block size and associativity etc., change the cache.cfg file under the cacti folder and run the simulator with the parameter of 4bit width machine size. With the process, we will reach the RUU access time and cache latency. We treat the access time as the clock cycle time and then calculate cache latency using cache access time divide the RUU access time.

To be mentioned, we need to build the cache.cfg file for each simulation respectively. And the latency should be changed to the clock cycle, which is equal to the access time of cache respectively divided by RUU access time.

## MIPS rates for the two applications for the different machine widths in a table and in a graph: 1 pt.

With the configuration file I get above, using sim\_outorder to get the performance of the different setting of the machine and calculate the MIPS as following:

Using instructions per cycle to calculate the MIPS, and the RUU access time to determine the processor frequency.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1/1/1/1 | 2/2/2/2 | 4/4/4/4 | 8/8/8/8 |
| Perl(MIPS) | 560 | 654 | 611 | 508 |
| Compress95(MIPS) | 681 | 866 | 766 | 641 |

## Total transistor counts and areas for the different machine widths in a table and in a graph: 1 pt

The transistor and the areas are get by the estimator excel file by modifying the parameters in the first sheet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1/1/1/1 | 2/2/2/2 | 4/4/4/4 | 8/8/8/8 |
| Transistor counts | 346384607 | 346528582 | 346816532 | 347392432 |
| Areas | 155805548828 | 155969084476 | 156325963388 | 157158951676 |

## Explanations as described in section IV above: 0.5 pt.

With the increment of Fetch/Decode/Issue queue the area and transistor number will increase with the same trend, but will not change dramatically (still in the same order of magnitude).

To generate the most desirable machine we should consider both area, transistor usage, and MIPS. Base on the above chart listed, the 2/2/2/2 machine size get me the best MIPS with the simulation of the two provided files. Also, with the help of Real Estate Estimator, the 2/2/2/2 is still a good tradeoff between to get the best performance with relatively small area and the number of transistors.