Assignment

Project name	SA2		
Document ref			
Version			
Release date			
Author	Wu Fei		
Classification	[Document classification]		
Distribution List	[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

1 Processing

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

```
ee557@ee557:~/Desktop/p2p3$ sim-outorder -dumpconfig /home/ee557/Desktop/p2p3/my_configuration
sim-outorder: SimpleScalar/PISA Tool Set version 3.0 of August, 2003.
Copyright (c) 1994-2003 by Todd M. Austin, Ph.D. and SimpleScalar, LLC.
All Rights Reserved. This version of SimpleScalar is licensed for academic
non-commercial use.  No portion of this work may be used by any commercial
entity, or for any commercial purpose, without the prior written permission
of SimpleScalar, LLC (info@simplescalar.com).
```

```
ee557@ee557:~/Desktop/p2p3$ vim my_configuration
ee557@ee557:~/Desktop/p2p3$ ls
BenchMarks_Little  cacti65  my_configuration  spec95-little  test.bin.little
ee557@ee557:~/Desktop/p2p3$
```

2. Adding constrain for both simulation files:

```
# maximum number of inst's to execute
-max:inst                       15000000

# number of insts skipped before timing starts
-fastfwd                        5000000

# generate pipetrace, i.e., <fname|stdout|stderr> <range>
# -ptrace                   <null>

# instruction fetch queue size (in insts)
"test_perl" 162L, 4436C written                                         41,41         16%
```

```
# maximum number of inst's to execute
-max:inst                       50000000

# number of insts skipped before timing starts
-fastfwd                        5000000

# generate pipetrace, i.e., <fname|stdout|stderr> <range>
# -ptrace                   <null>

# instruction fetch queue size (in insts)
"test_compress" 162L, 4436C written                                     41,41         16%
```

3. Using Real Estimator to get the entry size and some other parameter we need
4. Calculate the access time with the parameter we get in 4
5. Get the latency needed by the configuration file we generator in 1
6. Complete the configuration file and using SimpleScalar Simulator to calculate the processor performance
7. 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:

```
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:19 mytest_compress95
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:45 mytest_compress95_w1
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:52 mytest_compress95_w2
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:57 mytest_compress95_w8
-rw-rw-r-- 1 ee557 ee557    4439 Oct  9 11:18 mytest_perl
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:46 mytest_perl_w1
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:53 mytest_perl_w2
-rw-rw-r-- 1 ee557 ee557    4438 Oct  9 11:57 mytest_perl_w8
```

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

2 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.

2.1 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.

2.2 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

Level 1 i-cache		
Level 1 i-cache cell height	40.0	SRCellBasicHghtInLam + (Lev1ICacheReadPort + Lev1ICacheWrtPort) * WtWidInLam
Level 1 i-cache cell width	56.0	SRCellBasicWidInLam + (Lev1ICacheReadPort + 2 * Lev1ICacheWrtPort) * WtWidInLam
Level 1 i-cache data area	293,601,280.0	Lev1ICacheDataBit * Lev1ICacheCellHght * Lev1ICacheCellWid
Level 1 i-cache tag area	13,189,120.0	Lev1ICacheTagBit * Lev1ICacheCellHght * Lev1ICacheCellWid
Total level 1 i-cache area	**306,790,400.0**	**Lev1ICacheDataArea + Lev1ICacheTagArea**

Level 1 d-cache		
Level 1 d-cache cell height	56.0	SRCellBasicHghtInLam + (Lev1DCacheReadPort + Lev1DCacheWrtPort) * WtWidInLam
Level 1 d-cache cell width	80.0	SRCellBasicWidInLam + (Lev1DCacheReadPort + 2 * Lev1DCacheWrtPort) * WtWidInLam
Level 1 d-cache data area	1,174,405,120.0	Lev1DCacheDataBit * Lev1DCacheCellHght * Lev1DCacheCellWid
Level 1 d-cache tag area	55,050,240.0	Lev1DCacheTagBit * Lev1DCacheCellHght * Lev1DCacheCellWid
Total level 1 d-cache area	**1,229,455,360.0**	**Lev1DCacheDataArea + Lev1DCacheTagArea**

Level 2 d-cache		
Level 2 d-cache cell height	56.0	SRCellBasicHghtInLam + (Lev2DCacheReadPort + Lev2DCacheWrtPort) * WtWidInLam
Level 2 d-cache cell width	80.0	SRCellBasicWidInLam + (Lev2DCacheReadPort + 2 * Lev2DCacheWrtPort) * WtWidInLam
Level 2 d-cache data area	150,323,855,360.0	Lev2DCacheDataBit * Lev2DCacheCellHght * Lev2DCacheCellWid
Level 2 d-cache tag area	2,789,212,160.0	Lev2DCacheTagBit * Lev2DCacheCellHght * Lev2DCacheCellWid
Total level 2 d-cache area	**153,113,067,520.0**	**Lev2DCacheDataArea + Lev2DCacheTagArea**

2.3 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

RUU	
RUU cell height	112.0
RUU cell width	160.0
RUU area	22,077,440.0

2/2/2/2

RUU	
RUU cell height	136.0
RUU cell width	192.0
RUU area	32,169,984.0

4/4/4/4

RUU	
RUU cell height	184.0
RUU cell width	256.0
RUU area	58,032,128.0

8/8/8/8

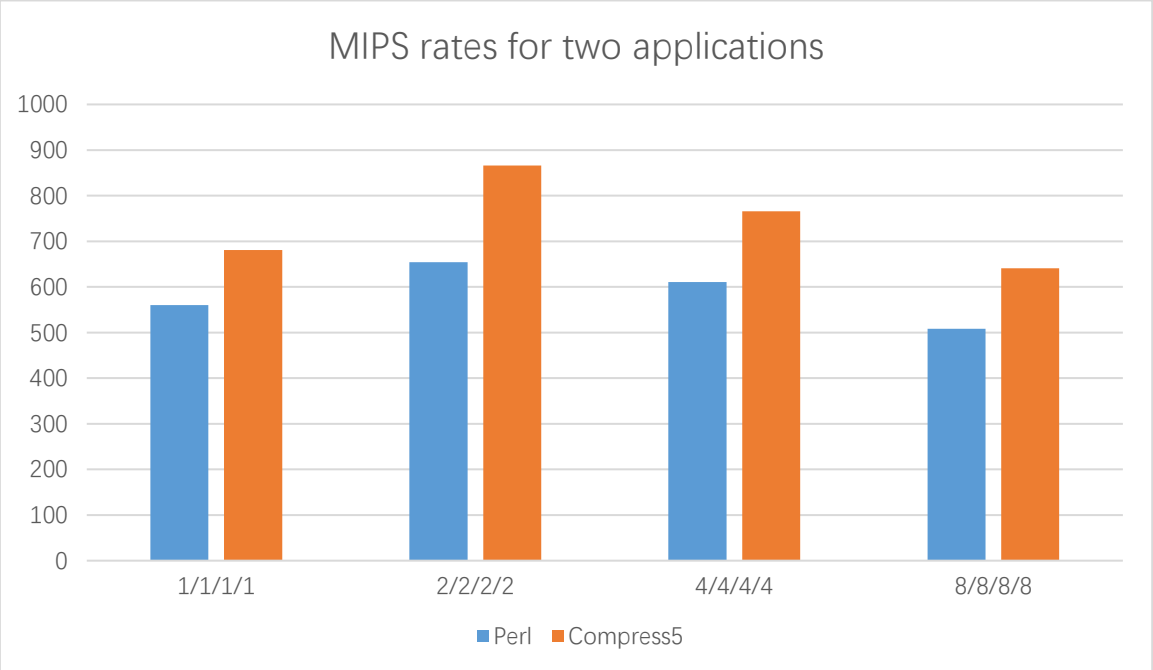
RUU	
RUU cell height	280.0
RUU cell width	384.0
RUU area	132,464,640.0

2.4 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.

2.5 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.

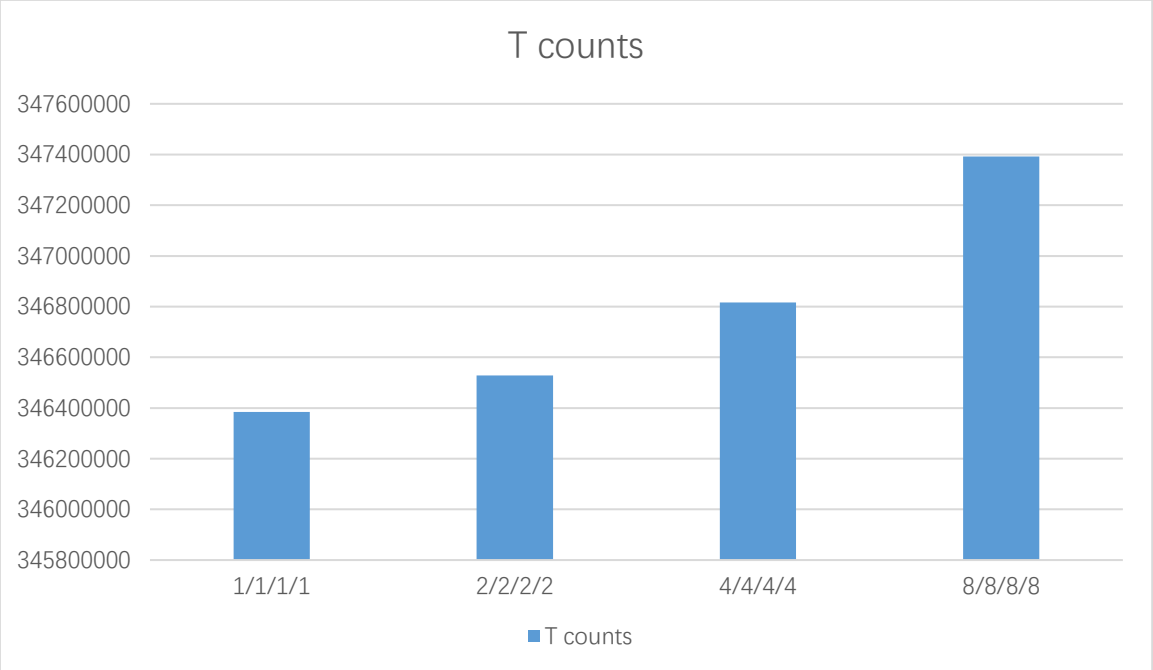


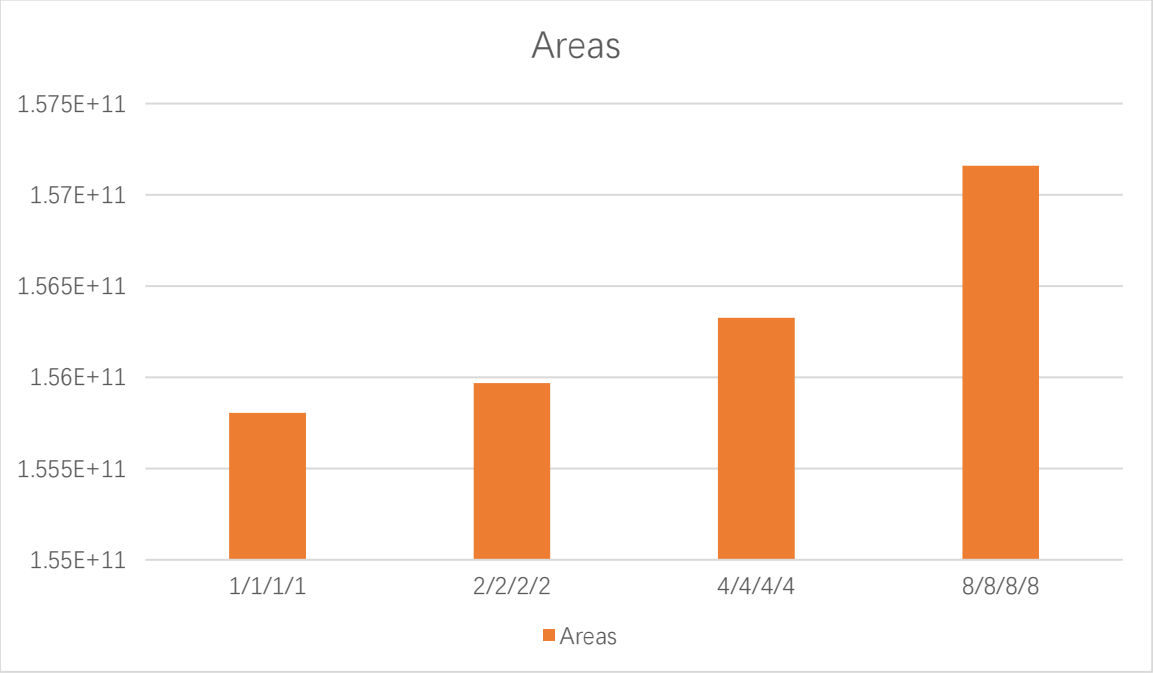
	1/1/1/1	2/2/2/2	4/4/4/4	8/8/8/8

Perl(MIPS)	560	654	611	508
Compress95(MIPS)	681	866	766	641

2.6 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

2.7 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.