School of Engineering and Computer Science

NWEN 242 **Computer Organization**

Project 3

The objective of this lab is to enhance your understanding of caches. It is worth 10% of your final grade. The project is marked out of 100.

Contents:

1. Introduction

In this lab, you will need to complete five exercises and to answer a number of related questions. For this purpose, you will use a new simulator called **SPIM-CACHE** that simulates instruction and data caches of a MIPS processor. These caches can be:

- Direct-Mapped,
- 2-Way Set Associative,
- 4-Way Set Associative, or
- Fully Associative.

They can have block sizes of 1W, 2W, or 4W. Also, data cache write policy can be either write-through or write-back.

A tutorial of using SPIM-CACHE is downloadable from the course web page. SPIM-CACHE is a simulator that runs MIPS32 assembly language programs. SPIM-CACHE permits the simulation of a data cache, an instruction cache, or both. To display the contents of the cache or caches being simulated, the user interface contains two special frames, namely the INSTRUCTION CACHE frame and the DATA CACHE frame, as shown in figure 1. To control the cache simulation, you may select the *Cache Simulation* option in the *Settings* dialog, which pops up after clicking on the *Settings* entry of the *Simulator* menu (see figure 2).

You can also control cache configuration through the *Cache Settings* dialog, shown in figure 3. This dialog can be activated by clicking the *Cache Settings* entry of the *Cache Simulation* menu. It allows you to choose your cache configuration, that is, cache size, block size, mapping functions, etc. If the *Show Rate* option is selected, some statistics are displayed in a small frame below the cache frame. For the data

cache, the number of misses is broken down according to the three categories, i.e., Compulsory, Capacity, and Conflict.

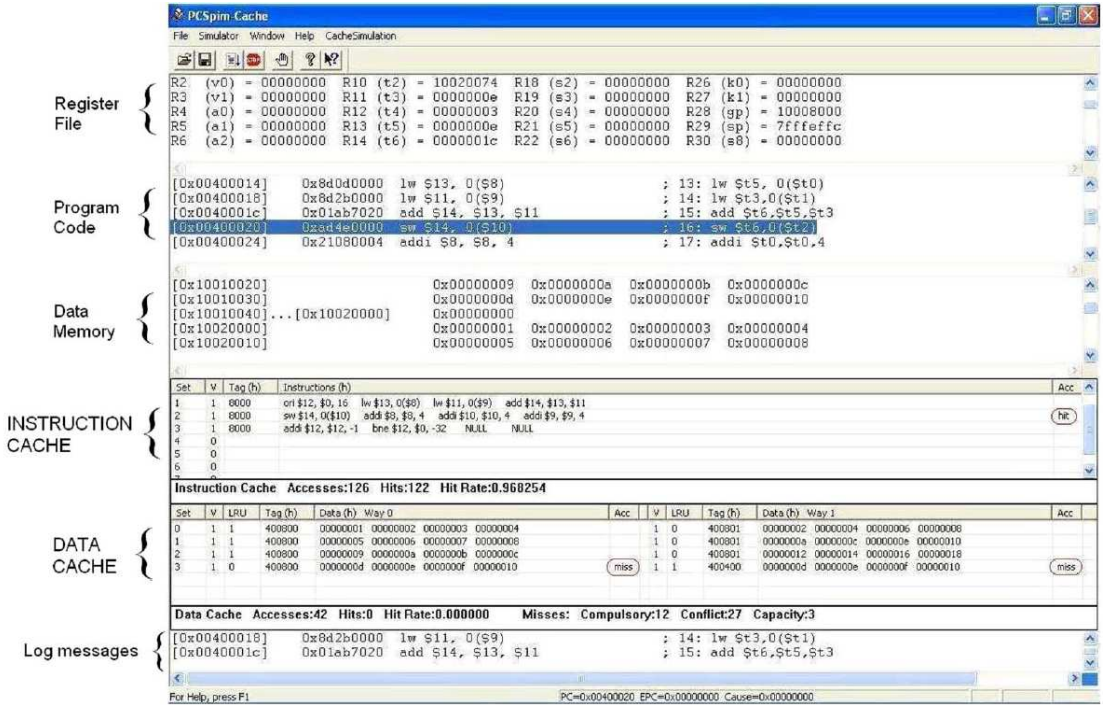

Figure 1: Main interface of SPIM-CACHE.

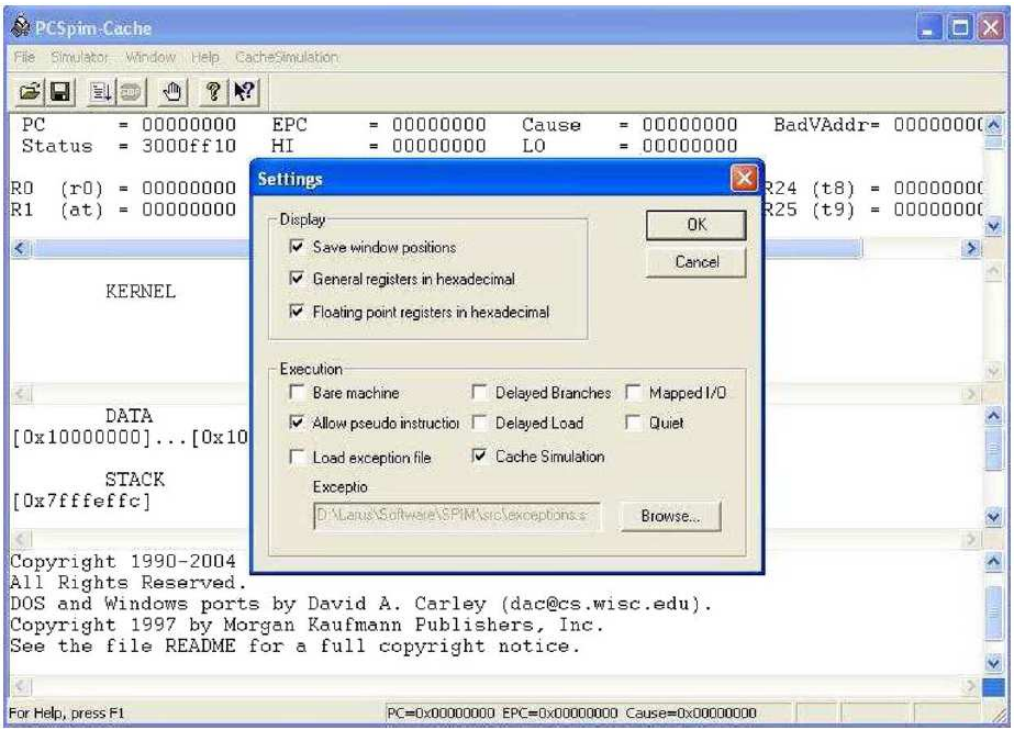

Figure 2: Cache simulation option.

1.1 How to start SPIM-CACHE

To start SPIM-CACHE, you can download the exe file (i.e. spim-cache_080605.exe) from the course web page. If you have a Windows machine, you can double click on the exe file in the Windows machine to start SPIM-CACHE. If you are using a Linux computer in the lab, please follow the instructions below:

Step 1: Click on the KMenu button  .

Step 2: In the search box of the popup menu, enter "**ward**".

Step 3: Click on the search result with the name **ward**, a remote desktop to the **ward** will be opened.

Step 4: Login to the **ward** using your ECS user name and password.

Step 5: Click the "Start" button in the Windows environment.

Step 6: Click the option "spim-cache" in the popup menu.

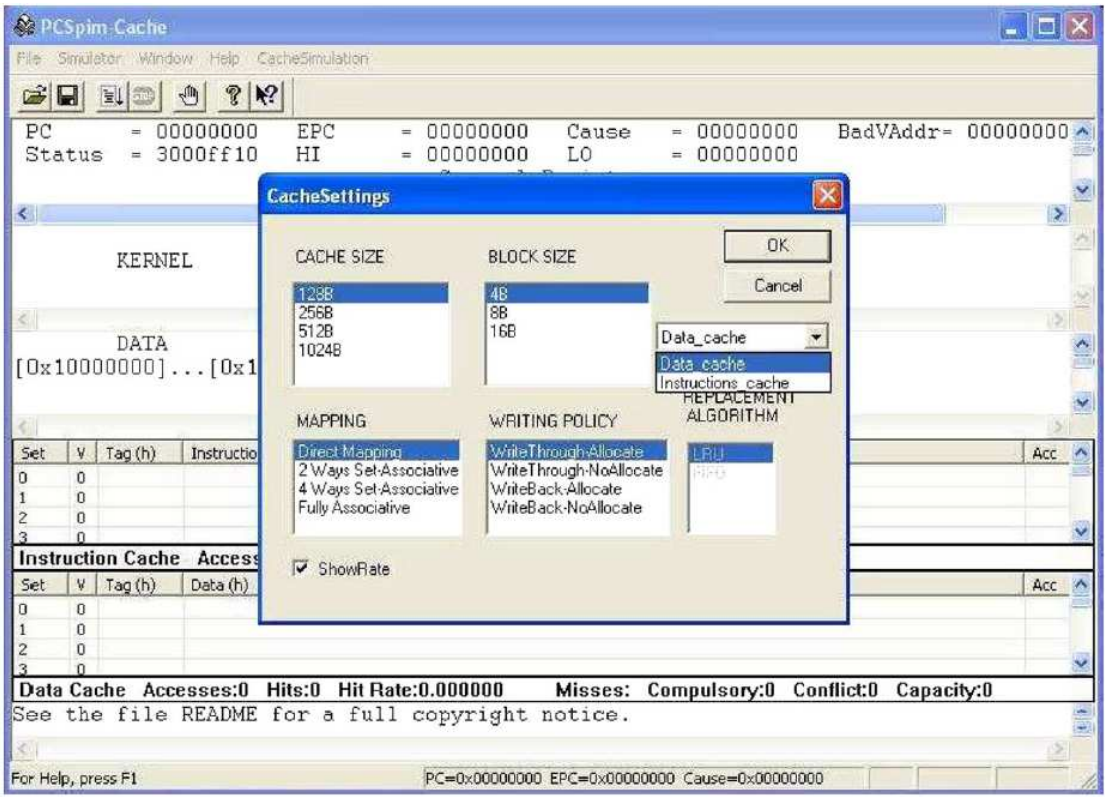

Figure 3: Cache settings dialog box.

2. How to prepare for the lab

1. Attend the lectures on Caches and Memory Hierarchy.
2. Read the relevant chapter in the textbook (Chapter 5).
3. Answer the preparatory questions.

3. Preparatory questions

Give short answers to the following questions.

a) What is a cache memory?

b) How does a cache memory improve the performance of a processor?

c) What are cache hits and cache misses?

d) What happens when a processor's request can't be satisfied from the cache?

e) Which types of cache memories did we consider in lectures?

f) How does the cache block size influence the processor's performance?

g) Given a direct mapped cache of the size 32 words with a block size of 4 words and a memory address 00000000 00000000 11000000 01000110, show the slot number, the word offset, and the tag value.

h) Show memory addresses of the words stored in a 4 W block size cache that contains the address 00000000 00000011 00001001 10001011.

i) Describe the Write-Through and Write-Back policy.

4. What you have to do to complete the lab

4.1 Mapping functions [30 marks]

In this exercise, you will work on how memory blocks map to cache. To perform this exercise, type and load the code in table 1 into SPIM-CACHE.

sum = 0;
for(i=0;i<8;i++) {
sum = sum + Array_A[i];
}
	.data 0x10000480
	Array_A:
	.word 1,1,1,1,2,2,2,2
	Array_B:
	.word 3,3,3,3,4,4,4,4
	.text
	.globl __start
	__start:
	la $2, Array_A
	li $6, 0 #sum=0
	li $4, 8 #number of elements
	loop: lw $5, 0($2)
	add $6, $6, $5 #sum=sum+Array_A[i]
	addi $2, $2, 4
	addi $4, $4, -1
	bgt $4, $0, loop
	.end

Table 1: Algorithm and corresponding MIPS code to study mapping functions.

(a) Load the program in table 1 into SPIM-CACHE. Configure the cache settings to be 256-B size, 16-B line size, and four-way set associative. Run the program and record the results (i.e. the status of each register) [7 marks].

(b) Note down the contents of data cache after the program is completed. Explain in detail how the elements of Array_A are mapped to each slot in the data cache [7 marks].

(c) Note down the contents of the instruction cache after the program is completed. Explain in detail how the contents are obtained through executing the program [7 marks].

(d) Extend the program in Table 1 to implement "sum=∑(Array_A[i]+Array_B[i])" . Execute the extended program based on the cache settings of "256-B size, 16-B line size, four-way set associative". After the program finishes, record the results (i.e. the status of each register). Also note down the contents of the data cache and explain in detail how the elements of Array_A and Array_B are mapped to each slot in the data cache [9 marks].

4.2 Temporal and spatial locality [20 marks]

(a) Based on the cache settings of "256-B size and four-way set associative", run the code that you have developed in 4.1(d) with three different block sizes for the cache

- 16B
- 8B
- 4B

Note down the hit rate and classify the misses with respect to each setting of the line size. Explain in detail why a hit rate of 0.75 can be obtained after running your program when the block size is 16B. Based on the data recorded, present an analysis regarding the relationship between block size and spatial locality [10 marks].

(b) Write a MIPS program that implements the algorithm shown in table 2. Configure the cache settings to be 256-B size, 16-B block size, and four-way set associative. Run the program with different settings of N: N=1,5,10, and 100. For each setting of N, note down the corresponding hit rate and classify the misses. Explain in detail the results obtained and draw conclusions about the impact of the temporal locality on the cache performance [10 marks].

```
sum = 0;
for(j=0; j<N; j++) {
        for(i=0; i<8; i++) {
                sum = sum + Array_A[i] + Array_B[i];
        }
}
```

Table 2: An algorithm for studying temporal locality in c.

4.3 Replacement algorithms [20 marks]

In this exercise, you will study replacement algorithms. To perform this exercise, type and load the code in table 3 into SPIM-CACHE.

```
sum = 0;
for(j=0; j<N; j++) {
        for(i=0; i<128; i=i+stride) {
                sum = sum + Array_A[i];
        }
}
```

```
        .data 0x10000000
Array_A:
        .word 0, 1, 2, …, 126, 127   # 128 elements in total
        .text
        .globl __start
__start:
```

	li $8, 1 # $8 = iterations of external loop
	li $3, 1 # $3 = stride in elements
	li $6, 0 # $6 = sum
	sll $9, $3, 2 # $9 = stride in bytes = $3x4
	ext_loop:
	li $5, 128 # $5 = number of elements of Array_A
	li $4, 0 # $4 = index
	int_loop:
	lw $7, Array_A($4)
	add $6, $6, $7
	add $4, $4, $9 # address of next element
	sub $5, $5, $3
	bgt $5, $0, int_loop
	addi $8, $8, -1
	bgt $8, $0, ext_loop
	.end

Table 3: Algorithm and corresponding MIPS code for studying replacement algorithms and strides.

(a) Configure the cache settings to be 256-B size, 16-B block size, and direct mapped. Run the program in table 3 to completion. Identify the content of set 0 of the data cache after the program execution. Explain in detail why the content of set 0 is as identified after the program completes [6 marks].

(b) Configure the cache settings to be 128-B size, 16-B block size, and two-way set associative. Set the replacement algorithm to LRU, record how the LRU counter values of set 0 of the data cache change throughout the running of the program. Explain the changes of the LRU counter values based on the LRU replacement algorithm [7 marks].

(c) Follow the same cache settings as in 4.3(b) and set the replacement algorithm to FIFO. Record and explain how the content of set 0 of the data cache changes throughout the running of the program [7 marks].

4.4 Accessing at different strides [20 marks]

Accessing the elements of an array at different strides helps to acquire a solid understanding on cache memories.

(a) Modify the MIPS program shown in table 3 so that the program will ask users to enter a stride value and store the value entered in register $3 [6 marks].

Hint: to read an integer entered by user and store it in register $t0, use the code fragment below:

```
li $v0, 5
syscall
move $t0, $v0
```

(b) Configure the cache settings to be 256-B size, 16-B block size, and four-way set associative. Run the modified program with different settings of stride (i.e. stride=1, 2, 4, 8, 16, 32, and 64) and different settings of N (i.e. N=1, 5, 10, 100). Record the hit rate and classify the misses for each run of the program [7 marks].

(c) Explain the data recorded in 4.4(b), draw conclusions about how different stride values and number of iterations of the external loop impact on the hit rate. Present detailed explanation of your findings [7 marks].

4.5 Instruction cache [10 marks]

Type and load the code in table 4 into SPIM-CACHE.

```
.data 0x10000000
Base:
.word 0, 10, 20
.text
.globl __start
__start:
la $9, Base
addi $3, $0, 3
addi $4, $0, 4
addi $6, $0, 3 #number of loops
loop:
addi $5, $5, 1 #i++
lw $8, 4($9)
lw $2, 8($9)
add $8, $8, $3
sw $8, 36($9)
add $2, $2, $4
sw $2, 40($9)
addi $3, $3, 1
addi $4, $4, 1
bne $5, $6, loop
.end
```

Table 4: MIPS code for studying instruction cache.

(a) Let the instruction cache size be 128B, identify

- The instruction cache mapping type
- The instruction cache block size, and
- The instruction cache replacement algorithm

that result in the highest instruction hit rate after running the program in table 4. **Note**: there can be more than one optimal configuration. You need to identify all configurations that give the optimal performance. [5 marks]

(b) Explain why the cache settings identified by you is suitable for the program in table 4. Among cache mapping type, cache block size, and cache replacement algorithm, which factor plays the decisive role and why. [5 marks]

5. Submission method

- Submit electronically before **Friday 17 October 2014 at 23:59**.
- Submit your answers to questions 4.1, 4.2, 4.3, 4.4, and 4.5. Include in your submission all .asm files that you have used for each exercise.