EEL 3801 - Computer Organization
Summer 2013
Project #3

Due date: 7/31/2013 (11:59 P.M.)

The project code and report must be submitted online through WebCourses as specified.

1. Introduction

In this project you will be using MARS simulator to run your assembly language program and data cache hit/miss analysis. The simulator is available from the following URL: http://courses.missouristate.edu/KenVollmar/MARS/

2. An Illustrative Example

Consider a 4x10 array of numbers, each occupying one word, is stored in main memory locations 7A00 through 7A27 (hex). The elements of this array, A, are stored in column order. Assume a cache size of 8 blocks where each block contains only one word and word addressable memory. We show the cache contents for various iterations of i and j of the following program:

```
SUM=0
for  j=0 to 9 DO
   SUM=SUM+A(0,j)
end
AVE=SUM/10
for  i=0 to 9 do
   A(0,i)=A(0,i)/AVE
end
```

(a) Direct-mapped cache

Block position	Contents of data cache after pass:					
	$j = 9$	$i = 1$	$i = 3$	$i = 5$	$i = 7$	$i = 9$
0	A(0,8)	A(0,0)	A(0,2)	A(0,4)	A(0,6)	A(0,8)
1						
2						
3						
4	A(0,9)	A(0,1)	A(0,3)	A(0,5)	A(0,7)	A(0,9)
5						
6						
7						

(b) Associative-mapped cache

Block position	Contents of data cache after pass:			
	$j = 9$	$i = 0$	$i = 5$	$i = 9$
0	A(0,8)	A(0,8)	A(0,8)	A(0,6)
1	A(0,9)	A(0,9)	A(0,9)	A(0,7)
2	A(0,2)	A(0,0)	A(0,0)	A(0,8)
3	A(0,3)	A(0,3)	A(0,1)	A(0,9)
4	A(0,4)	A(0,4)	A(0,2)	A(0,2)
5	A(0,5)	A(0,5)	A(0,3)	A(0,3)
6	A(0,6)	A(0,6)	A(0,4)	A(0,4)
7	A(0,7)	A(0,7)	A(0,5)	A(0,5)

(c) Set-associative-mapped cache

	Block position	Contents of data cache after pass:			
		$j = 9$	$i = 3$	$i = 7$	$i = 9$
Set 0	0	A(0,8)	A(0,2)	A(0,6)	A(0,6)
	1	A(0,9)	A(0,3)	A(0,7)	A(0,7)
	2	A(0,6)	A(0,0)	A(0,4)	A(0,8)
	3	A(0,7)	A(0,1)	A(0,5)	A(0,9)
Set 1	0				
	1				
	2				
	3				

3. Data Cache Simulator tool for MIPS

A helpful document to get familiar with **Data Cache Simulator** in MARS simulator for MIPS programming is available at the following link:
http://faculty.kfupm.edu.sa/coe/aimane/ics233/ics233(062)lab-14.pdf

4. Project Description

Consider the following segment of a C++ code for multiplication of two matrices A and B of size (NxN) each:

```
for (int i=0;i<N;i++) {
   for (int j=0;j<N;j++) {
      temp = 0;
      for (int k=0;k<N;k++) {
         temp = temp + A[i][k]*B[k][j];
      }
      C[i][j] = temp;
   }
}
```

Thus, matrix C is the product of matrix A and matrix B. For example, for N=3:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix}, \qquad B = \begin{bmatrix} 4 & 5 & 6 \\ 4 & 5 & 6 \\ 4 & 5 & 6 \end{bmatrix}$$

The result C=AxB is given by:

$$A = \begin{bmatrix} 24 & 30 & 36 \\ 24 & 30 & 36 \\ 24 & 30 & 36 \end{bmatrix}$$

Write an assembly program for matrix multiplication where all the elements are integers and the matrices are stored in main memory.

1) Initially, hard code the input matrices A and B of size 3x3 each in your program by storing into main memory. You can choose either storing them in row-major or column-major order. Observe the cache performance (e.g., hit-rate, miss-rate etc.) for your code.
2) **Optimized Matrix Multiplication for N=3**: Optimize your code (by storing matrices in a particular order, code-reordering etc.) and cache organization (block-size, placement policy etc.) to get the maximum hit-rate for a fix cache size of 128 bytes. **Submit** the optimized mult3.asm file mentioning the optimized cache organization for N=3 in your report. Give a comprehensive description of how you improved the hit-rate.
3) Verify the cache hit-rate that you observed from the simulator by your hand-calculations.
4) **Matrix Multiplication for $N \geq 8$:** Show the hit-rate of your program for various cache parameters by extending the concept to N≥8 (Choose any number greater

than or equal to 8, e.g. 8x8 matrices A,B,C). Complete the following table by listing the hit-rates.

Cache-organization	Cache Block Size	Hit-rate
Direct Mapping	2	
	4	
	8	
Fully Associative	2	
	4	
	8	
2-Way Associative	2	
	4	
	8	

5) **Optimized Matrix Multiplication for N=8:** Show the optimized cache parameters to achieve the best possible hit-rate for this problem. **Submit** the optimized mult8.asm file mentioning the optimized cache organization for N=8 in your report. Give a description of how the hit-rate was improved.

Moreover, observe the following requirements overall:

- You are free to choose the data input method, e.g., by keyboard, file, or fixing in your assembly code. However, the matrices should be stored in main memory instead of just storing in the registers.
- A fix **Cache size=128** bytes should be used for all the above mentioned tasks.
- Although you can display the values etc. on console using syscall for debugging purpose, your final code should not contain any display type or other redundant instructions.
- For associativity >1, assume LRU (Least Recently Used) policy.

Turn in the written project report using the format given.

5. Project Report Format

Name:
Date:
Introduction(5 pts): An introduction explaining the lab assignment and what are the goals of the assignment (what is the code implementing).
Test Procedure (15 pts): This is a step by step procedure that tells how to test each element of the mult3.asm program. It must be detailed enough so the user can follow each step. This is similar to the steps in a cookbook. Include the screenshots of your program's output as well as the data cache performance window.
Solutions to the assigned tasks in sub-sections 4.2, 4.4, 4.5 as listed above in the project description section (30+20+30pts): Include the parameters in tabulated forms.

6. What you need to submit?

1. Project pdf report in the format as described above.
2. Commented assembly code files **mult3.asm** and **mult8.asm** for the program written for the project.