School of Computer Science
Georgia Institute of Technology

CS6290, Fall 2009
Hyesoon Kim, Instructor
Minjang Kim, Nick Black, TAs

Exam 1, October 1, 2009

Name :______________________________

GTID :______________________________

Problem 1 (15 points):________

Problem 2 (10 points):________

Problem 3 (10 points):________

Problem 4 (15 points):________

Problem 5 (10 points):________

Problem 6 (10 points):________

Total (70 points):____________

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is recorded on each sheet of the exam.

GOOD LUCK AND HAVE A GOOD FALL BREAK!

Name:________________________________

Problem 1 (15 points):

Part a. (5 points) What is a delayed branch? Discuss pros and cons.

Delayed branch is a technique to minimize the control hazard. One or more instructions following a delayed branch are always to be executed regardless of the delayed branch's direction which gives a processor to time to calculate the target address of the branch instruction. Pros: It reduces the penalty of branch execution when there is no branch prediction mechanism. Cons: The number of delayed slots (how many instructions will be always executed) must be known to ISA, which cannot be easily changed later. Hence, when a processor increases the pipeline depth, the actual time to calculate the target address and the number of delayed slots could be different which causes a complex hardware. The compiler often cannot find enough number of instructions to fill the delayed slot so it inserts no-op.

Part b. (5 points) Discuss why or when a global branch history (2-level branch predictor) can provide better performance than an 1 level branch predictor. Provide a code example.

Global branch history provides a path information which leads a branch. If a branch direction is dependent on (or correlated with) other previous branches, using global branch history often increases the prediction accuracy. For example,

```
if (aa > 0)
    bb = 0;
else
   bb = -10;
 ....
 if (bb => 0) {
   ...
 }
```

The second branch direction is strongly dependent on the first branch's direction.

Part c. (5 points) DRAM has a destructive read characteristic. What does the DRAM controller do because of the destructive read characteristic?

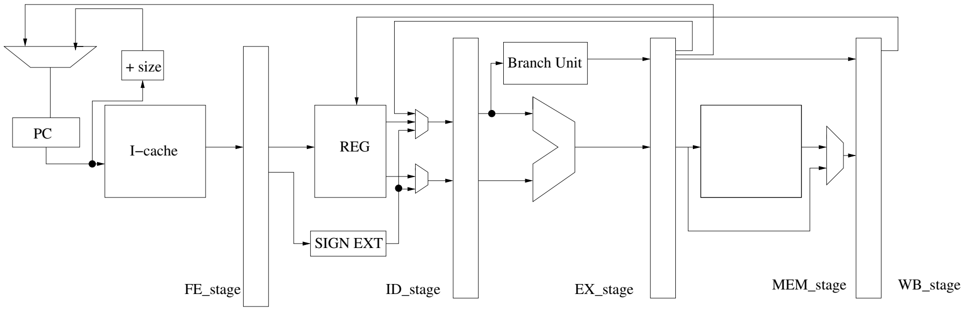
After the DRAM controller reads a value, it must write back the original value to the original location. To support that, the DRAM system has a row buffer.

Name:______________________________

Problem 2 (10 points):

The following Figure shows a pipeline design. The execution time of all instruction is one cycle except for multiplication (which takes 2 cycles) and division (which takes 4 cycles.) We assume that the register file can be written and read at the same cycle. How many cycles to complete the following sequence of instructions? We assume that the branch instruction at 0x08 is taken. Please show the work how you calculate the number of cycles. Note that a destination register is shown first. i.e., In (ADD R2, R1, R1)) R2 is the destination register id and R1s are the source register ids. In BR R4, TARGET, R4 is the source register. There is only one functional unit for ADD/SUB/MUL/DIV and none of the execution units is pipelined.

```
        0x00 ADD R1, R2, R3
        0x04 SUB R4, R1, R3
        0x08 BR  R4, TARGET
        0x10 MUL R2, R2, -1
        0x14 ADD R1, R7, R3
TARGET  0x18 ADD R5, R2, R3
        0x20 MUL R6, R2, -1
        0x24 ADD R7, R5, R2
```



Answer = 16 Cycles

Name:______________________________

Cycle	FE_stage	ID_stage	EX_stage	MEM_stage	WB_stage
1	ADD				
2	SUB	ADD			
3	BR	SUB	ADD		
4	BR	SUB		ADD	
5	MUL	BR	SUB		ADD
6	MUL	BR		SUB	
7	MUL		BR		SUB
8	MUL			BR	
9	ADD				BR
10	MUL	ADD			
11	ADD	MUL	ADD		
12		ADD	MUL	ADD	
13		ADD	MUL		ADD
14			ADD	MUL	
15				ADD	MUL
16					ADD
17					
18					
19					
20					
21					
22					
23					
24					
25					
26					
27					
28					
29					
30					

Note that the table is provided for your work. The table shows instructions inside the pipeline stages (not the outcome of the latch.)

Name:______________________________

Problem 3 (10 points):

Part a. (5 points) We like to find out execution latency of each operation in a new CPU. We wrote several micro-benchmarks that have different instruction mixtures and measure the execution time of each micro-benchmark. The processor has 2 GHz CPU frequency. Here are the results of the execution time and the instruction mixture. Find the CPI for each instruction category. To simplify the problem, we assume that all load/store instructions have the same CPI.

Benchmark	Integer/BR	FP	Load/Store	execution time	# of instructions
Benchmark 1	60%	0%	40 %	3.2 msec	1M
Benchmark 2	40%	40%	20 %	3.4 msec	1M
Benchmark 3	30%	30%	40 %	3.8 msec	1M

Instruction	CPI
Integer/BR	4
FP	8
Load/Store	10

Solve the following three equations
0.6*CPI_int + 0 * CPI_FP + 0.4*CPI_LS = 6.4
0.4*CPI_int + 0.4 * CPI_FP + 0.2*CPI_LS = 6.8
0.3*CPI_int + 0.3 * CPI_FP + 0.4*CPI_LS = 7.6

Part b. (5 points) Calculate the average CPI for each micro-benchmark and the average of all 3 benchmarks. What is the average of IPC for all micro-benchmarks?

Benchmark	CPI
Benchmark 1	6.4 =(3.2msec/(1/2G)/1M)
Benchmark 2	6.8 =(3.4msec/(1/2G)/1M)
Benchmark 3	7.6 =(3.8msec/(1/2G)/1M)
Average	6.934

Average IPC = 1/6.934 = 0.144

Name:________________________________

Problem 4 (15 points):

The following table shows a profiling result of a program.

FuncA	70%
FuncB	20%
FuncC	10%

Note that FuncA, FuncB, FuncC must be executed in sequential. Assume that if a parallelized section is executed on an n-core machine, the parallelized section is executed n times faster than the sequential version of the code. All the machines have IPC 1.

Part a. (5 points) After parallelizing only FuncA, we run the program on a 1GHz 256-core machine. What is the maximum speedup over 1GHz 1-core machine?

$1/(0.7/256+0.3\ ) = 3.30$

Part b. (10 points) Assume that only FuncA and FuncB are parallelized. Among the three machines in the below table, which one do you choose to run the program to get the best performance? Show the work.

Machine A	500 MHz 512-core
Machine B	2GHz 32-core
Machine C	4GHz 1-core and 1GHz 16-core machine

Note that for Machine C, you can use either 4GHz 1-core or 1GHz 16-core in a given time.

Compared to an 1GHz sequential machine, the speedup for each machine is
Speedup A = $1/(0.9*2/512+0.1*2) = 4.91$
Speedup B = $1/(0.9/(32*2)+0.1/(2)) = 15.6$
Speedup C = $1/(0.9/16 + 0.1/4) = 12.3$
Hence, the answer is Machine B

Name:________________________________

Problem 5 (10 points):

A byte-addressable computer has a 128B data cache. The cache block size is 128 bits. The cache uses the true LRU replacement policy. When a given program is executed, the processor reads data from the following sequence of hex addresses:

0x0500, 0x5004, 0x0508, 0x050C, 0x050A, 0x0500, 0x05F4, 0x0500, 0x0504, 0x0518, 0x051C, 0x054C, 0x05F4, 0x500C

Part a. (5 points) Compute the hit rate if a direct-mapped cache is used.

Addr	Index	Hit/miss
0x0500	0	M
0x5004	0	M
0x0508	0	M
0x050C	0	H
0x050A	0	H
0x0500	0	H
0x05F4	7	M
0x0500	0	H
0x0504	0	H
0x0518	1	M
0x051C	1	H
0x054C	4	M
0x05F4	7	H
0x500C	0	M

Hit rate 7/14 = 50%

Part b. (5 points) Compute the hit rate if a 2-way set-associative cache is used.

Addr	Index	Hit/miss
0x0500	0	M
0x5004	0	M
0x0508	0	H
0x050C	0	H
0x050A	0	H
0x0500	0	H
0x05F4	3	M
0x0500	0	H
0x0504	0	H
0x0518	1	M
0x051C	1	H
0x054C	0	M
0x05F4	3	H
0x500C	0	M

Hit rate = 8/14

Name:________________________________

Problem 6 (10 points):

A computer has a 8KB write back cache and 2MB physical memory. Each cache block is 4B, the cache is 4-way set associative and uses the true LRU replacement policy. Assume 32-bit address space for virtual address and byte-addressable memory. It uses a 4KB page size. The cache uses a virtual tag and physical index. How big (in bits) is the tag store? You do not need to worry about space for cache coherence state.

# of sets = (8KB/(4B*4)) = 2^9
Index bits = $log_2(2^9)$) = 9 bits Boffset bits = 2 bits (4B) Tag = 32-9-2 = 21 bits (virtual address space)
(21 + 1(dirty)+1(valid)+2(LRU))*4 (way) * 2^9 (# of sets)) = 50 Kbits