Full Name:____________________________________

Andrew ID (print clearly!):____________________________________

740: Computer Architecture, Fall 2013

Midterm II

November 25, 2013

Instructions:

- Make sure that your exam has 15 pages and is not missing any sheets, then write your full name and Andrew login ID on the front.
- This exam is closed book. You may not use **any** electronic devices. You may use one **single-sided** page of notes that you bring to the exam.
- Write your answers in the box provided below the problem. If you make a mess, clearly indicate your final answer.
- Be concise. You will be penalized for excessive verbosity. Use no more than 15 words per answer, unless otherwise stated.
- The exam lasts 1 hour 20 minutes.
- The problems are of varying difficulty. The point value of each problem is indicated. Do not spend too much time on one question. Good luck!

Problem	Your Score	Possible Points
1		55
2		28
3		30
4		30
5		42
6		20
Total		205

Please read the following sentence carefully and sign in the provided box:

"I promise not to discuss this exam with anyone until Wednesday, November 27."

Signature:

Problem 1: Potpourri (55 pts)

A) [10 pts] Thread prioritization

Suppose we are running a **multithreaded** application where threads are part of the same application on a multicore processor. The memory controller is shared between the cores.

1) Provide one reason why prioritizing a memory non-intensive thread over a memory-intensive one in the memory controller would improve performance. If this is not possible, write N/A and explain why.

2) Provide one reason why doing the same would degrade performance. If this is not possible, write N/A and explain why.

B) [4 pts] Memory bandwidth

Under what conditions would an application's performance increase linearly as memory bandwidth is increased?

C) [4 pts] Fat trees

What problem does the fat tree interconnect solve that is present in the tree interconnect?

D) [10 pts] Interconnect

You are observing a system with many processing elements connected through a network. There is currently no activity on the network (no messages are being sent). On cycle 10, one of the cores generates a message destined for a cache bank somewhere else on the network. You observe the network on cycle 20 and see that this message has not departed the source location. Assume that all components are enabled (not powered off) and operating at full speed. There are no other messages present in the system at this time. Why could this be?

Continued . . .

E) [12 pts] Slack

As you recall, we have discussed the idea of slack based prioritization for on-chip interconnects in class. In fact, you reviewed a paper that introduced this concept. The key idea was to prioritize the packet that has the least slack over others in the router, where the slack of a packet (ideally) is defined as the number of cycles the packet can be delayed without hurting performance.

The concept of slack is actually more general. It can be applied to prioritization at any shared resource, assuming the slack of a "memory request" can be estimated well.

1) Suppose we have a mechanism that tries to estimate the exact slack of a memory request when the request is injected into the shared resources. Provide two reasons why estimating the exact slack of a packet might be difficult:

2) What performance issue can slack-based prioritization cause to some processors in the system? Why?

3) How can you solve this problem?

F) [5 pts] Dataflow

What is the purpose of token tagging in dynamic dataflow architectures?

Continued . . .

G) [10 pts] Alpha 21264

The Alpha 21264 had a "Prefetch and evict next" instruction that "prefetched data into the L1 cache except that the block will be evicted from the L1 data cache on the next access to the same data cache set."

1) What access patterns could benefit from this instruction? Explain well.

2) The Alpha 21264 processor employed a predictor that predicted whether a load would hit or miss in the cache before the load accessed the cache. What was the purpose of using this predictor? Explain concisely but with enough detail.

Problem 2: Multithreading (28 pts)

Suppose your friend designed the following fine-grained multithreaded machine:

- The pipeline has 22 stages and is 1 instruction wide.
- Branches are resolved at the end of the 18th stage and there is a 1 cycle delay after that to communicate the branch target to the fetch stage.
- The data cache is accessed during stage 20. On a hit, the thread does not stall. On a miss, the thread stalls for 100 cycles, fixed. The cache is non-blocking and has space to accommodate 16 outstanding requests.
- The number of hardware contexts is 200.

Assuming that there are always enough threads present, answer the following questions:

A) [7 pts] Can the pipeline **always** be kept full and non-stalling? Why or why not? *(Hint: think about the worst case execution characteristics.)*

CIRCLE ONE: **YES** **NO**

B) [7 pts] Can the pipeline **always** be kept full and non-stalling if all accesses hit in the cache? Why or why not?

CIRCLE ONE: **YES** **NO**

C) [7 pts] Assume that all accesses hit in the cache and your friend wants to keep the pipeline **always** full and non-stalling. How would you adjust the hardware resources (if necessary) to satisfy this while minimizing hardware cost? You cannot change the latencies provided above. Be comprehensive and specific with numerical answers. If nothing is necessary, justify why this is the case.

D) [7 pts] Assume that all accesses miss in the cache and your friend wants to keep the pipeline **always** full and non-stalling. How would you adjust the hardware resources (if necessary) to satisfy this while minimizing hardware cost? You cannot change the latencies provided above. Be comprehensive and specific with numerical answers. If nothing is necessary, justify why this is the case.

Problem 3: Return of Tomasulo's Algorithm (30 pts)

The diagram below shows a snapshot at a particular point in time of various parts (reservation stations and register alias table) of the microarchitecture for an implementation supporting out-of-order execution in the spirit of Tomasulo's Algorithm. Note that there is an adder and a multiplier in this machine. The processor is supplied with a seven instruction program following reset. The state below was captured at some point in time during the execution of these seven instructions. Anything marked with a – is unknown and can't be relied upon for your answer. You should assume that the bottommost instruction in the reservation station arrived earliest and the topmost instruction in the reservation station arrived last.

	SRC 1			SRC 2		
ID	**V**	**Tag**	**Val**	**V**	**Tag**	**Val**
-	-	-	-	-	-	-
E	0	C	-	0	A	-
F	0	A	-	1	-	20
C	0	A	-	0	A	-

×

	SRC 1			SRC 2		
ID	**V**	**Tag**	**Val**	**V**	**Tag**	**Val**
-	-	-	-	-	-	-
D	0	B	-	0	E	-
B	1	-	20	0	F	-
A	1	-	20	1	-	30

+

RAT

Reg	V	Tag	Val
R0	1	-	5
R1	0	A	10
R2	0	B	20
R3	1	-	30
R4	0	C	40
R5	0	D	50
R6	1	-	60
R7	1	-	70

Continued . . .

A) [15 pts] Identify the instructions and draw the data flow graph for the seven instructions (use + for ADD and * for MUL). Please label the edges of the data flow graph with the destination register tag if known. Label with register number if the tag is not known. Note that the first instruction is an ADD with destination register R3.

B) [15 pts] Fill in the instruction opcodes, source, and destination registers in the table below.

Instructions

OP	DEST	SRC1	SRC2
ADD	R3		

Problem 4: Tiered-difficulty (30 pts)

Recall from your required reading on Tiered-Latency DRAM that there is a near and far segment, each containing some number of rows. Assume a very simplified memory model where there is just one bank and there are two rows in the near segment and four rows in the far segment. The time to activate and precharge a row is 25ns in the near segment and 50ns in the far segment. The time from start of activation to reading data is 10ns in the near segment and 15ns in the far segment. All other timings are negligible for this problem. Given the following memory request stream, determine the optimal assignment (minimize average latency of requests) of rows in the near and far segment (assume a fixed mapping where rows cannot migrate, a closed-row policy, and the far segment is inclusive).

```
time 0ns:  row 0 read
time 10ns:  row 1 read
time 100ns:  row 2 read
time 105ns:  row 1 read
time 200ns:  row 3 read
time 300ns:  row 1 read
```

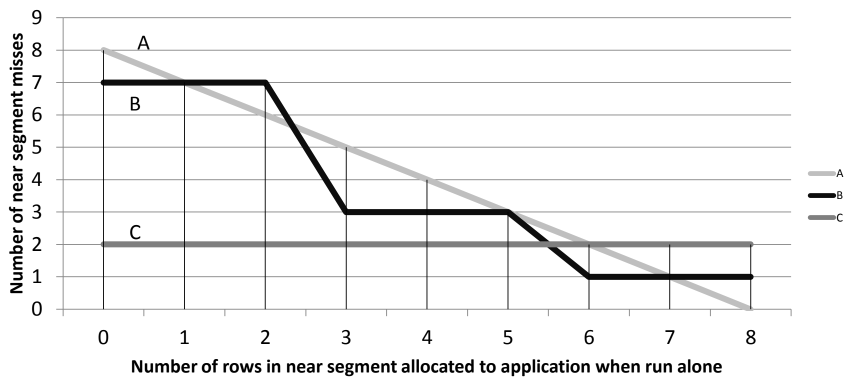
A) [6 pts] What rows would you place in near segment? Hint: draw a timeline.

B) [6 pts] What rows would you place in far segment?

C) [6 pts] In 15 words or less, describe the insight in your mapping?

D) [6 pts] Assume now that the mapping is dynamic. What are the tradeoffs of an exclusive design vs. an inclusive design? Name one advantage and one disadvantage for each.

E) [6 pts] Assume now that there are eight (8) rows in the near segment. Below is a plot showing the number of misses to the near segment for three applications (A, B, and C) when run alone with the specified number of rows allocated to the application in the near segment. This is similar to the plots you saw in your Utility-Based Cache Partitioning reading except for TL-DRAM instead of a cache. Determine the optimal static partitioning of the near segment when all three of these applications are run together on the system. In other words, how many rows would you allocate for each application? Hint: this should sum to eight. Optimal for this problem is defined as minimizing total misses across all applications.



1) How many near segment rows would you allocate to A?

2) How many near segment rows would you allocate to B?

3) How many near segment rows would you allocate to C?

Problem 5: GPUs (42 pts)

We define the **SIMD utilization** of a program running on a GPU as the fraction of SIMD lanes that are kept busy with active threads during the run of a program.

The following code segment is running on a GPU. Each thread executes a single iteration of the shown loop. Assume that the data values of the arrays A, B, and C are already in vector registers so there are no loads and stores in this program. (Hint: Notice that there are 5 instructions in each thread as labled below.) A warp in the GPU consists of 64 threads, and there are 64 SIMD lanes in the GPU.

```
for (i = 0; i < 16384; i++) {
  if (A[i] > 0) {           //Instruction 1
    A[i] = A[i] * C[i];     //Instruction 2
    B[i] = A[i] + B[i];     //Instruction 3
    C[i] = B[i] + 1;        //Instruction 4
    D[i] = C[i] * B[i];     //Instruction 5
  }
}
```

A) [2 pts] How many warps does it take to execute this program?

B) [10 pts] As shown below, assume array A has a repetitive pattern which has 32 ones followed by 96 zeros repetitively and array B has a different repetitive pattern which has 64 zeros followed by 64 ones repetitively. What is the SIMD utilization of this program?

A:	1	1	...29 1s...	1	0	0	...93 0s...	0	...32 1s...	96 0s...	...
B:	0	0	...61 0s...	0	1	1	...61 1s...	1	...64 0s	...64 1s...	...

C) [10 pts] Is it possible for this program to yield a SIMD utilization of 25%?

CIRCLE ONE: **YES** **NO**

If YES, what should be true about arrays A and B for the SIMD utilization to be 25%? Be precise and show your work. If NO, explain why not.

D) [10 pts] Is it possible for this program to yield a SIMD utilization of 20%?

CIRCLE ONE: **YES** **NO**

If YES, what should be true about arrays A and B for the SIMD utilization to be 20%? Be precise and show your work. If NO, explain why not.

E) [10 pts] During an execution with a particular input array A, which has exactly 24 positive elements in every 64 elements, Hongyi finds that the SIMD utilization of the program is 50%. Based on this observation, Hongyi claims that any input array that has an **average** of 24 out of 64 elements positive would yield a 50% SIMD utilization. Is Hongyi correct?

CIRCLE ONE: **YES** **NO**

If YES, show your work. If NO, provide a counterexample.

Problem 6: Hyperblock (20 pts)

As described in class, Hyperblock scheduling uses predication support to replace unbiased branches with predicates, which enables larger code blocks.

A) [2 pts] In one sentence, in terms of code optimizations, explain what benefit does larger scheduling code blocks provide?

One optimization that can be applied to Hyperblock is **Instruction Promotion**. Instruction Promotion hoists the operation from a predicated instruction and replaces the original predicated instruction with a conditional move. With Instruction Promotion, operations can be scheduled and issued before their corresponding predicates are determined. Below shows an example of Instruction Promotion.

Before:

```
        cmplt B6,B7-> B0
[B0]    ld MEM1-> A5
[!B0]   ld MEM2-> A5
        nop 4
        addi A5,8 -> A8
```

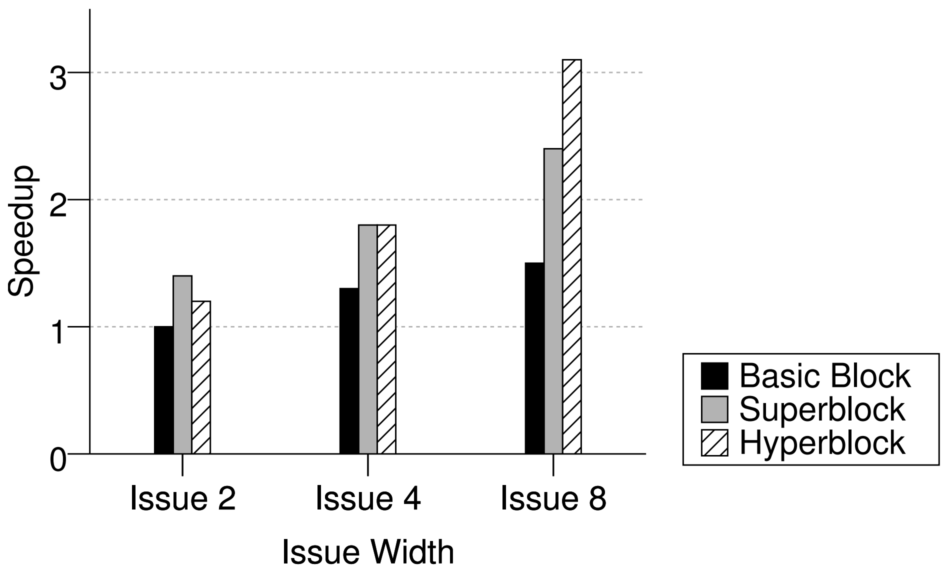
After Instruction Promotion:

```
        ld MEM1-> A5
        ld MEM2-> A6
        cmplt B6,B7-> B0
[!B0]   mv A6-> A5
        nop 4
        addi A5,8 -> A8
```

Assume we run this code on a processor that supports predication but can only issue a predicated instruction after its corresponding predicate has been resolved.

B) [4 pts] For the example above, can Instruction Promotion ever improve system performance? Why or why not?

C) [4 pts] For the example above, can Instruction Promotion ever degrade system performance? Why or why not?



D) [10 pts] The graph above shows the performance comparison of a program optimized using Hyperblock and Superblock respectively with different issue widths. With all other factors being equal, as the figure shows, when the issue width is low, Superblock provides higher speedup than Hyperblock. However, when the issue width is high, Hyperblock provides higher speedup than Superblock. Explain why this can happen?

Scratch Paper

Scratch Paper