ECE411 Computer Organization and Design
Final Exam
12/14/2018

NetID:____________

Exam Guidelines:
1. This exam has **6 problems.** Make sure you have a complete exam before you begin **[19 pages + the cover page]**.
2. Write your name on every page in case pages become separated during grading.
3. You will have **3 hours** to complete this exam.
4. Write all of your answers on the exam itself. If you need more space to answer a given problem, continue on the back of the page, but clearly indicate that you have done so.
5. This exam is closed-book. You may use one sheet of notes. You may use a calculator.
6. **DO NOT** do anything that might be perceived as cheating. The minimum penalty will be a grade of zero.
7. Show all of your work on all problems. Correct answers that do not include work demonstrating how they were generated may not receive full credit, and answers that show no work cannot receive partial credit.
8. The exam is meant to test your understanding. Ample time has been provided. So be patient and read the questions/problems carefully before you answer.
9. **Good luck!**

Problem	Points
1. **32** pt	
2. **33** pt	
3. **23** pt	
4. **20** pt	
5. **28** pt	
6. **23** pt	
Total: **159** pt	

Name:__________________________

Question 1: Caches

After finishing your pipelined MP3 processor, you decide that you'd like to work on something more challenging. You want to create a multiple-issue machine by duplicating your pipeline (now there are two copies of the in-order pipeline). You decide to implement a unified L1 data cache to avoid maintaining coherence between two private caches. Once you've made this decision, you realize that you'll need a data cache which can service accesses to two arbitrary addresses in a given cycle (ideally) and decide to build a **two-bank data cache with two access ports** to serve this purpose. You will design your pipelines such that instruction pairs will move through the pipeline in lock-step where pipeline 0 will execute the first instruction in program order and pipeline 1 will execute the second.

A multi-banked cache is a cache configuration where the entire address space is partitioned into sets by using *clog2(#Banks)* bits from the address to determine the bank that a given address maps to. Each bank will have its own storage arrays for tags, data, and metadata. By having multiple banks of independent storage arrays, it's possible to have multiple access ports to the cache. When the access ports are servicing addresses which map to different banks, the requests can be serviced together in the same cycle. In the event that multiple requests are made to the same bank, there exists a *bank conflict* and the requests must be serialized.

In your processor, addresses are byte-addressable and 32-bits in length, cache lines are 64B, and the word size is 4B. You will implement a true LRU policy (requiring N*clog2(N) bits) for any configuration which is not direct-mapped. Your metadata must consist of valid, dirty, and true LRU bits. Your cache will support writes of any alignment with the only guarantee being that accesses will not span multiple cache lines. For the following questions, you can assume a hit rate of 100%.

a. Given that your cache has **two** identical banks, write an equation for the total number of bits needed by your cache (tags + data + metadata) if the configuration of each bank can vary in the number of sets and the number of ways. (8 pts)

b. To partition the memory address space into two disjoint sets, one bit from the address must be used to decide the bank mapping. If it is known that for a given application, two simultaneously executing memory accesses (one in each pipeline) will access cache lines which are separated by a stride of four, which bit ([31:0]) of the address should be used to select the bank in order to provide the highest performance? Explain your reasoning. (8 pts)

c. In the event of a *bank conflict*, the most naive solution is that accesses to the cache must be serialized to ensure correct execution. If the two addresses being accessed map to different sets in the same bank, the accesses cannot be serviced together since the storage arrays can only read out a single set each cycle. On the other hand, if the addresses map to the same set, the accesses may or may not require serialization. If there are two accesses which map to the same cache line, describe one scenario which requires serialization and one scenario which does not. If serialization is required, in which order should the cache service the requests? (8 pts)

d. Unhappy with the performance penalties of naively serializing all accesses which result in bank conflicts, you decide to improve performance by adding an optimization to your cache. Describe such an optimization which can allow your cache to resolve a previously serialized pair of memory accesses in one cycle. The scenario you optimize may be the same as you found in part c so long as it can be resolved in one cycle using added logic. (8 pts)

Name:________________________

Question 2: OoO Execution

Consider a load-store queue with the following properties

- Each entry in the store queue has an address and some data, each of which may either have some concrete value (given as hexadecimal "0x____") that is available immediately, or some physical register (given as "PR___") which will become available in some later cycle. Similarly an entry in the load queue will have an address, but instead of data, it has a destination register also given as "PR___"
- A store is "committed" in the cycle where the address and data become available.
- Loads and stores must go through the cache in program order. If the head of the store queue has its address and data available but follows an entry in the load queue that has not yet been freed, then the store queue must wait.
- Assume that when a value for some physical register become available on the bus, it is not at the beginning of the cycle, so the value may be loaded into a queue entry in that cycle, but cannot be sent to the cache until the beginning of the next cycle.
- Assume that for the store queue if the address and data are available at the beginning of the cycle, then the data cache will respond in the affirmative (100% hit rate) by the end of that same cycle. The corresponding entry is then free at the beginning of the next cycle.
- For example if the head of the store queue has its address available but is waiting on the data to be stored, and that data becomes available in cycle 14, then we say it was committed in cycle 14, in cycle 15 the data and address are given to the cache and the cache responds by the end of the cycle, and the entry in the queue is freed in cycle 16.
- Similarly for the load queue, if the address is available at the end of cycle 14, it will interact with the cache in cycle 15 and a result will be available at the end of that cycle, which will be broadcast on the bus in cycle 16.

Fill in the blanks (10 pts):

Name:___________________________

Store Queue:

Entry	Address	Data	Follows	Cycle Committed	Cycle Freed
1	PR15	PR13	-		
2	0x1A58	0xB0AR	1		
3	PR7	0x5678	1		

Load Queue:

Entry	Address	Follows	Destination	Cycle Broadcast	Value Broadcast
1	0xB0	1	PR1		
2	PR30	3	PR23		
3	PR6	3	PR16		

Register values broadcast from other execution units:

Register	Becomes available in cycle	With value
PR15	2	0x1A58
PR7	4	0xD860
PR6	6	0xB21C
PR30	8	0x1A58
PR13	10	0xBAAD

Memory Contents:

Memory Address	Contents
0xA0	0x600D
0xB0	0xBEAD
0xC0	0x7EEA
0xD0	0xC0FE

Name:________________________

2) Repeat the exercise with the following modifications (10 pts):

- If two entries in the store queue have the same address and of the loads that fall in order between the stores, there are no unresolved addresses and no address that have the same address, the oldest store will be removed.
- If an entry in the load queue falls in order behind an entry in the store queue with the same address and the data in that entry is available and no store falling in order between the two entries has an unresolved address, the entry will be removed from the load queue and the data broadcast as the result of that load.
- When the conditions for the removal of a load or store are met at the beginning of a cycle, the change will be reflected at the beginning of the next cycle (which will be the cycle we say the store is freed or the load broadcast).
- A store that is not at the head of the queue may write to memory out of order if it has its address and data resolved and all preceding loads and stores have resolved their address and those addresses do not match the address of the store. Of course if multiple stores are able to be written to memory, we give priority to the store that comes earliest in the ordering.
- A load that is not at the head of the queue may read from memory ahead of preceding loads with unresolved addresses if it has its address available and all preceding stores have resolved addresses.
- When the conditions are met for a load or store to jump the queue (described in the last two bullet points) at the beginning of a cycle, the load or store may access memory in that same cycle and will be freed/broadcast in the next cycle.

Name:___________________________

Store Queue:

Entry	Address	Data	Follows	Cycle Committed	Cycle Freed
1	PR15	PR13	-		
2	0x1A58	0xB0AR	1		
3	PR7	0x5678	1		

Load Queue:

Entry	Address	Follows	Destination	Cycle Broadcast	Value Broadcast
1	0xB0	1	PR1		
2	PR30	3	PR23		
3	PR6	3	PR16		

Register values broadcast from other execution units:

Register	Becomes available in cycle	With value
PR15	2	0x1A58
PR7	4	0xD860
PR6	6	0xB21C
PR30	8	0x1A58
PR13	10	0xBAAD

Memory Contents:

Memory Address	Contents
0xA0	0x600D
0xB0	0xBEAD
0xC0	0x7EEA
0xD0	0xC0FE

3) In the exercises above, we used two approaches to load/store address disambiguation. Describe another approach that may reduce average instruction latency and discuss the tradeoff between the costs and benefits of this approach (consider all dimensions – power/energy, design complexity, area cost, latency) (8 pts).

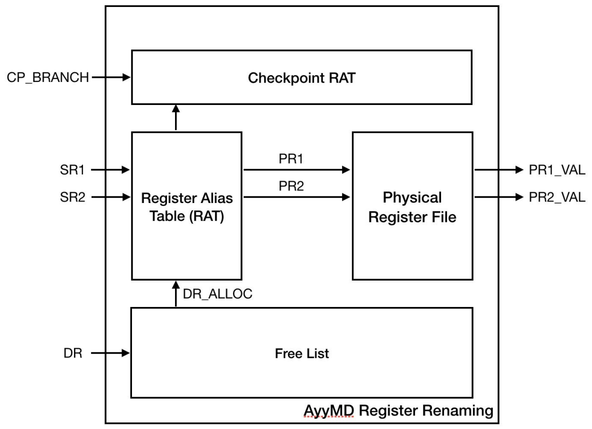
4) Rewrite the following sequence of instructions to manage address disambiguation in software (5 pts).

```
st r4, 0(r2)
ld r1, 0(r3)
```

Name:_________________________

Question 3: Pipelining & Control Hazards

You are working at a processor company called AyyyMD and are tasked with designing the register renaming for a new processing core. AyyyMD has decided to work on two processor cores, one with Simultaneous Multithreading (SMT) and one without. You are working on the SMT core's explicit register renaming. The processors have identical architectural specifications (Execution units, physical registers, etc.) with the only difference being the hardware support required for SMT. To improve reuse of hardware intellectual property (IP), you are told to use the register renaming block from the non-SMT core as a starting point. You are given the simplified microarchitecture block diagram of the renaming block.



DR_ALLOC - Physical destination register to allocate
PR - Physical register
SR - Source (architectural) register
CP_BRANCH - Signal to checkpoint RAT on branch

a. How can you take the existing register renaming hardware block and modify it to support SMT without changing architectural parameters? (Hint: What must be duplicated?) (8 pts)

You are asked to evaluate potential CPU designs for a new LIT64 based CPU complex to go into the SoC for the new DankPhone.

CPU Complex 1: Single core, dual issue, SMT, 32kB L1-I cache, 32kB L1-D cache, 128kB unified L2 cache, 2k entry 2-level branch predictor

CPU Complex 2: Dual core, single issue, no SMT, 16kB L1-I cache (per core), 16kB L1-I cache (per core), 128kB unified and shared L2 cache, 1k entry 2-level branch predictor (per core)

You decide to run a couple of benchmarks to try and determine which core is optimal in terms of performance. Each benchmark consists of two similar threads. You find that benchmark 1 runs faster on CPU complex 1. However, you find that benchmark 2 runs faster on CPU complex 2.

b. Give and explain a reason why CPU complex 1 completed benchmark 1 faster than complex 2. (5 pts)

Name:__________________________

c. Give and explain a reason why CPU complex 2 completed benchmark 2 faster than complex 2. (5 pts)

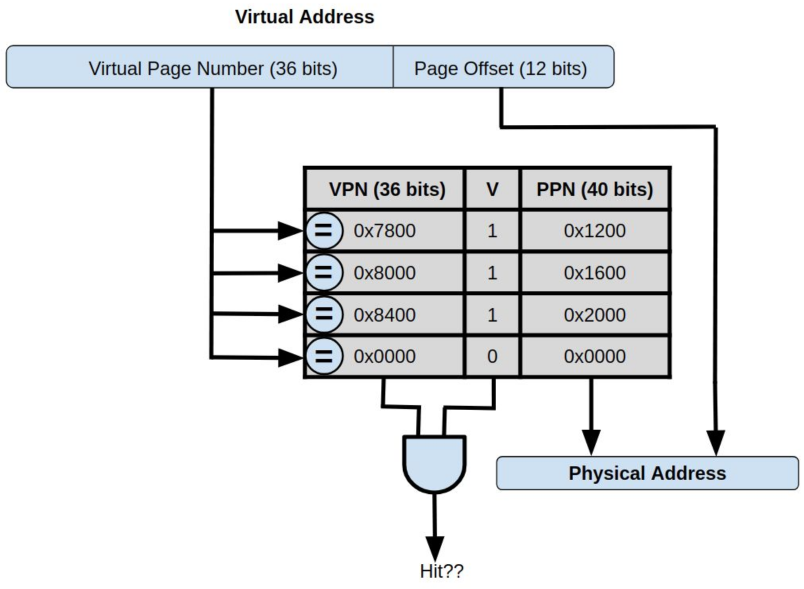
d. In a superscalar, out-of-order CPU with explicit register renaming and precise exception handling, name 3 possible structural hazards. (5 pts)

Name:___________________________

Question 4: System Level Consideration

After the huge success of the "*deep.SHALLOW*"[1] architecture on the mobile platforms, "*LittleHope*" has decided to move the architecture to the server platforms. This requires significant modifications to the existing architecture.You have been assigned to lead that project:

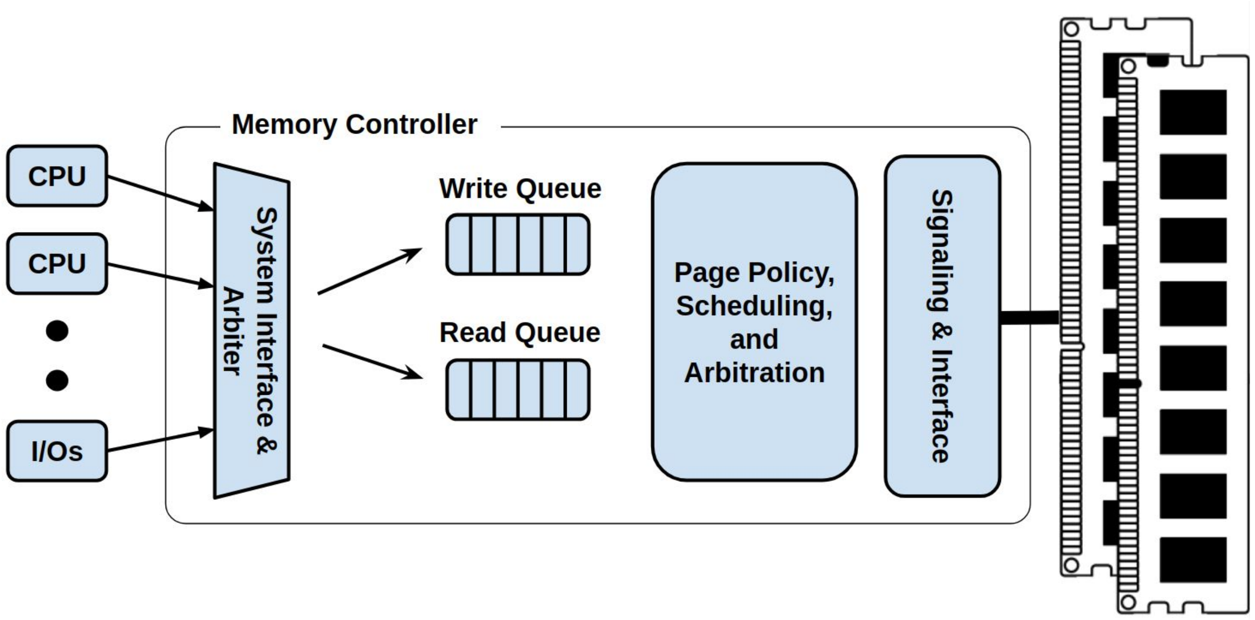
1. One of the required modifications is supporting different page sizes in the TLBs for the new architecture. For the mobile platforms, the architecture supported only a single page size (4KB) but the new architecture should support 4KB, 2MB, and 1GB page sizes. The figure below shows the TLB that is used in the mobile platforms. Does it require any modifications to benefit from supporting multiple page sizes? If yes, state your suggested changes to the TLB. (12 pts)
   [Note: In "*deep.SHALLOW*", any data in the DRAM cannot be mapped to two different page table entries with two different page sizes at the same time.]



[1] An architecture couples shallow, in-order pipeline depth cores (*SHALLOW*) with deep, in-order pipeline depth cores (*deep*) on the same chip.

Name:___________________________

2. The figure below depicts the new memory controller that one of your teams has implemented. The scheduling policy prioritizes Reads over Writes. The verification team reported an issue while working on the new design.
   The problem is observed while reading a cache line which was updated and evicted from the caches back to the memory recently. The controller returns stale data for the previously evicted line. What do you think causes the bug? How do you propose to fix it? (8 pts)



Name:________________________

Question 5: Cache Coherency

Firefly

DEC Firefly was a multiprocessor machine designed by computer scientists who left Xerox to form their own company in Palo Alto, CA in 1984. The Firefly cache coherence scheme was a relatively simple design with its own strengths and weaknesses.

Firefly Specifications:

1. Firefly uses a snoop-based write-update scheme that propagates writes to other caches and main memory when the line is in a shared state.
2. If the cache line is the only copy, updates will not be propagated.
3. The coherence bus has mechanisms for informing processors/caches that the accessed line is shared/not-shared. Transactions may utilize this information and you may indicate it using SL (Shared Line), NSL (Not-Shared Line).
4. Each cache will put a "read-miss", "write" or "write-miss" transaction on the bus when servicing a request to inform the other caches. These bus transactions can be snooped.

Coherence States and Explanations	Processor Transactions	Bus Transactions
VE - Valid Exclusive: Only copy and clean VS - Valid Shared: Shared copy and clean. D - Dirty: Only copy and dirty.	P-Read-SharedLine P-Read-NotSharedLine P-Write-SharedLine P-Write-NotSharedLine	B-Read-Miss B-Write B-Write-Miss

Name:__________________________

a. Given the information about the Firefly cache coherence scheme, draw the Firefly state diagram. You do not need to show unrealistic cases such as a snooped-hit when the line is in an exclusive state. (12 pts)

b. Consider a program with multiple threads running across two processor cores. The two cores are operating on a large shared dataset. Core 0 is the producer of data and executes mostly store operations. Core 1 is the consumer of data and executes mostly reads. Both of the cores use the entirety of the dataset. Will the Firefly coherence scheme perform better in this scenario than an invalidation-based coherence scheme such as the MSI protocol you saw in class? Why? (8 pts)

Coherence Interactions with Speculation

c. Consider an out-of-order processor with support for speculative execution and MSI cache coherence. To ensure correctness, you know that the architectural state of a processor (registers, memory) should not be changed until the speculative state is resolved. For higher performance, this processor accesses the cache speculatively. If the access is a miss, the cache will begin fetching the missed line immediately.

To eliminate any adverse effects this may cause on other cores, to what state should the speculatively loaded cache line be initialized and kept in if the serviced miss is for a speculative store operation? What coherence and memory traffic will occur once the speculative state is resolved to complete the store? Make sure to think about the state of the same cache line in other cores, and explain your reasoning. (8 pts)

Name:_________________________

Question 6: Potpourri:

1. What optimizations may not be applied in a processor that is expected to guarantee sequential consistency: a) write buffering, b) non-blocking reads, c) compiler-based code-ordering, d) all of them. Why? (5 pts)

2. You choose to build a machine that supports an ISA with a single instruction:

```
subleq a, b, c   ; Mem[b] = Mem[b] - Mem[a]
                 ; if (Mem[b] ≤ 0) goto c
```

Conditional branching can be suppressed by setting the third operand equal to the address of the next instruction in sequence. If the third operand is not written, this suppression is implied.

How would you emulate MOV a, b with a set of subleq instruction? (5 pts)

3. Profiling of a large number of memory accesses of a program revealed that stream1 to DRAMs was processed considerably faster than stream 2.

stream1: 0, 2048, 4096, 6144, 8192…
stream2: 0, 4096, 8192, …

What could be a likely explanation?(5 pts)

Name:_______________________

4. Modern processors processor 32, 64, etc., bits at a time. You colleague designed a bit serial processor that processes one bit at a time. The bit serial processor was found to have *increased* energy over the baseline designs? Why? (5 pts)

5. For the latest episode of "Fun with Flags", $heldon and eh-Me asked for suggestions for a flag for ECE411. Draw one and add a tagline. (3 pts)