2012/Spring Computer Architecture PhD Qualifying Exam

Student ID _____ Name _____

1. [15pts] Suppose you want to design a cache with 128KB data capacity. Your architecture uses 32-bit address space. For the following configurations, calculate the width of tag (in bits) for each cache-line. Please, explain how you find the answer. Do not just write the final answer.
   - 8-way associative, block size = 32B
   - Direct-mapped, block size = 128B
   - 16-way associative, block size = 4B

2. [10pts] An ideal pipelined design can have the speed up of N from the single cycle design, when N is the number of pipeline stages. However, in reality, the speedup can be less than N. List all possible reasons why the pipeline may not achieve the ideal speedup?

3. [20pts] Dependency and scheduling

```
X1:  add r1, r2, r3
X2:  or  r3, r2, r1
X3:  sub r2, r4, r5
X4:  sw  r2, 0(r6)
X5:  and r2, r2, r7
X6:  lw  r4, 0(r7)
```

   A. Find all ***possible*** dependencies
   B. What is the minimum number of cycles to run the instructions with an ideal out-of-order processor? The execution of an instruction takes one cycle, and the unlimited number of instructions can be executed in parallel, if the instructions do not have dependencies. *However, register renaming is not supported.*
   C. What is the minimum number of cycles if unlimited register renaming is supported? Use the same assumption as 3-B

4. [20pts] The problem assumes the following system configurations
   32-bit architecture, which uses 4GB address space for each process
   Page size: 4KB page, the size of each page table entry : 4B

   A. What is the page table size for each process for one-level page table (flat page table)?
   B. What are the minimum and maximum page table sizes for a process, if the system uses two-level page tables? (For the minimum case, a process uses the memory which fits in one page.)
   C. To hide the access latency for TLBs, TLBs may be accessed in parallel with accesses to L1 caches. To support such parallel accesses to TLBs and L1 caches, the organization of L1 caches may be restricted to meet certain conditions. If the maximum associativity is limited to 8 ways, what is the largest cache capacity for such L1 caches? (a physical address must be mapped to only one set in the cache)

D. Suppose the maximum capacity from the 4.C was X bytes. What if you want to have a cache with 2X bytes capacity? You still want to access TLBs and L1 caches in parallel. What problems will occur and how will you solve the problem?

5. [20pts] The following code fragment is a spin lock implementation written by a newly hired programmer. It spins until a variable (pointed by R1) becomes zero, and if the variable is zero, it is locked by setting it to 1.

```
Lock acquire:
                li R2,#1
    lockit:     lw R3,0(R1)
                bnez R3,lockit
                sw R2,0(R1)
```

A. Explain why the above code fragment does not implement a spin lock correctly. Describe a scenario when the above implementation can allow multiple threads to enter the critical section

B. Another programmer fixed the above bug, and implemented a new spin lock with atomic swap instruction (exch). Describe a scenario when the above spin lock can lead to a performance problem with many unnecessary invalidations. Re-write the code to fix the performance problem.

```
            li R2,#1
lockit:     exch R2,0(R1)       ;atomic exchange
            bnez R2,lockit
```

6. [15pts] Cache coherence

State	CPU		Snoop transition (by Bus command)	
	Command	Next state (Action)	Read	Next state (Action)
Invalid	Read	Shared (BusRead)	BusRead	Invalid
	Write	___(1)___ (BusRfo)	BusRfo	Invalid
			BusUp	Invalid
Shared	Read	___(2)___	BusRead	__(3)____
	Write	___(4)___ (__(5)___)	BusRfo	__(6)____
			BusUp	Invalid
Modified	Read	__ (7)____	BusRead	__(8)____ (WriteBack)
	Write	Modified	BusRfo	Invalid (WriteBack)
			BusUp	Error

BusRfo: Bus Read for Ownership

A. The complete above table to implement a simple MSI protocol for snoop-based coherence. (1-8)

B. Explain why “Modified” state cannot receive “BusUp” command. (“Error” state in the table)