CSE 141 Summer 2016 Homework 5

PID: _____

Name:_____

1. Assume that you have a computer with 4KB pages and a 4-entry full-associative TLB that uses LRU replacement policy. If page must be brought into main memory, increment the largest page number. If the current TLB content is

	Valid	tag	Physical page number
0	1	0xB	12
1	1	0x7	4
2	1	0x3	6
3	0	0x4	9

and the current page table is

	Valid	Physical page or in disk
0	1	0x5
1	0	Disk
2	0	Disk
3	1	0x6
4	1	0x9
5	1	0xB
6	0	Disk
7	1	0x4
8	0	Disk
9	0	Disk
10	1	0x3
11	1	0xC

Please identify how many TLB misses and page faults in the following address stream: 0x123D, 0x8B3, 0x365C, 0x871B, 0xBEE6, 0x3140, 0xC049

Answer:

Assume index 0 is the most recent used entry.

		TLB {TAG [PPN]} | PT update
0x123D	PF	**1[D]**, B[C], 7[4], 3[6]
0x08B3	M	**0[5]**, 1[D], B[C], 7[4]
0x365C	M	**3[6]**, 0[5], 1[D], B[C]
0x871B	PF	**8[E]**, 3[6], 0[5], 1[D]
0xBEE6	M	**B[C]**, 8[E], 3[6], 0[5] |1[D]
0x3140	H	**3[6]**, B[C], 8[E], 0[5]
0xC049	PF	**C[F]**, 3[6], B[C], 8[E]

3 Page faults and 3(6) TLB Miss // Page faults happens after TLB misses

2. Assume the virtual address space of the computer is 64 bits, each page is 8KB in size, each page table entry occupies 8 bytes memory and the system is running 6 processes concurrently.

   (1) If the computer uses conventional page table, what's the total size of page tables in the system?

   VA space: 2^{64}

   Page size: 2^{13}

   Page table entries (per process): $2^{64} / 2^{13} = 2^{51}$

   Total Page Tables size = #process * Page table entries * entry size

   $= 6 * 2^{51} * 8$

   $= 3 * 2^{55}$ bytes (96PB)

   (2) If we are building a 4-way set associative, virtually indexed, physically tagged cache, what's the maximum available cache size?

   A = 4

   lg(S) + lg(B) = 13

   C <= ABS = 4 * 8K = 32KB

3. You are building a system around a single-issue in-order processor running at **2 GHz** and the processor has a base CPI of 1 if all memory accesses are hits. The only instructions that read or write data from memory are loads (20% of all instructions) and stores (5% of all instructions). The processor uses virtually-indexed, physically-tagged caches with no penalty in address translation if the TLB access is a hit. However, if the TLB misses, the system needs 120ns to finish the address translation and TLB updates. The TLB miss rate is 2%. The L1 cache is split into I-cache and D-cache with no penalty on hits. Both the I-cache and D-cache are direct mapped and hold 32KB each. You may assume the caches use write-allocate and write-back policies. The L1 I-cache has a 2% miss rate and the L1 D-cache has a 5% miss rate. Also, 50% of all blocks replaced from L1 D-cache are dirty. The 512KB write-back, unified L2 cache has an access time of 10ns. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to main memory. Also 25% of all blocks replaced are dirty. The main memory has an access latency of 60ns. What is the overall CPI, including memory accesses?

Cycle time = 1 / 2Ghz = 0.5 ns
TLB miss: 120ns = 240 cycles
L2 access: 10ns = 20 cycles
Memory access: 60ns = 120 cycles

Instruction Type	Percent	CPI Influence factors
Other Instruction	75%	① ③
Load Instruction	20%	① ② 2x③
Store Instruction	5%	① ② 2x③

CPI Influence factors:
① I cache miss
② D cache miss
③ TLB miss

L2 miss penalty = Memory access + L2 write back penalty
= 120 + (5% + 20%) * 120
= 150 cycles

L1-I miss penalty = L2 access latency + L2 miss rate * L2 miss penalty
= 20 + (1-80%) * 150
= 50 cycles

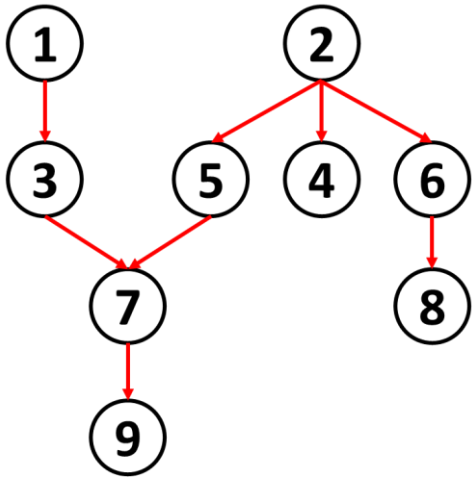
L1-D miss penalty = L2 access latency + L2 miss rate * L2 miss penalty
+ L1 write back penalty
= (20+0.2*150)*(1+50%)
= 75 cycles

Overall CPI = Basic CPI + ① CPI due to I cache miss +
CP I due to D cache miss ② CPI due to TLB misses ③
= 1 + 2% * 50 + (20% + 5%) * 5% * 75
+ 75% * 2% * 240 + 2 * (20% + 5%) * 2% * 240
= 8.94

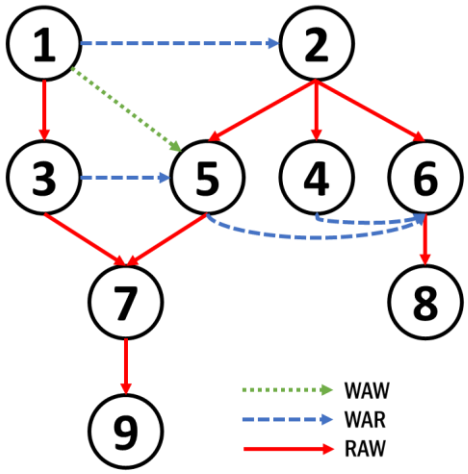
5. Consider the following execution sequence of MIPS instructions:

```
1: LOOP:   lw     $t1, 0($a0)
2:         addi   $a0, $a0, 4
3:         add    $v0, $v0, $t1
4:         bne    $a0, $t0, LOOP
5: LOOP:   lw     $t1, 0($a0)
6:         addi   $a0, $a0, 4
7:         add    $v0, $v0, $t1
8:         bne    $a0, $t0, LOOP
9:         sw     $v0, 0($a1)
```

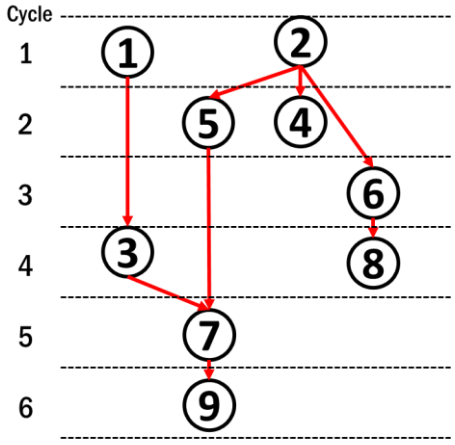
(1) Please draw the data dependency graph



(2) Please identify the false dependencies (WAW and WAR) in the given code



(3) The processor that you're given is a 2-issue out-of-order processor with unlimited physical registers and it takes 3 cycles to finish a load/store, 1 cycle to finish other instructions and with a perfect branch predictor. Assume all instructions are now already in the instruction window. How many cycles it takes to issue (moving out from schedule to execution) all the dynamic instructions?



It takes 6 cycles to issue all instructions.

6. Consider the following two processors:

   I. CPU CMP: 2-core superscalar processor with out-of-order issue capabilities. Each core has two functional units and these functional units can be pipelined if it needs more cycles to execute. Only a single thread can run on each core at a time.

   II. CPU SMT: An SMT processor that allows instructions from two threads to be run concurrently on the two functional units, and instructions from either or both threads can be issued to run on any cycle.

   Assume we have two threads A and B to run on these CPUs that include the following operations (You may assume these processors have full capability of eliminating false dependencies and all the instructions are already in instruction queues. If not specified, the instruction will take one cycle to execute):

Thread A	Thread B
A1: takes 2 cycles to execute A2: depends on the result of A1 A3: conflicts for a functional unit with A2 (They must use the same one) A4: depends on the result of A2	B1: no dependencies B2: conflicts for a functional unit with B1 (They must use the same one) B3: no dependencies B4: depends on the result of B2

(1) How many cycles will it take to issue these two threads on each processor?

CPU CMP

Core #1		Core #2	
A1	A3	B1	B3
		B2	
	A2	B4	
	A4		

4 cycles for thread A

3 cycles for thread B

CPU SMT

A1	A3
B1	B3
B2	A2
B4	A4

4 cycles for thread A

4cycles for thread B

(2) How many issue slots are wasted due to hazards?

For CPU CMP: 8 slots

For CPU SMT: 0 slots