CSE 141 Spring 2016 Homework 5

PID: A11489111

Name: Amit Nijjar

1. Consider the following matrix transpose code

```
int i, j ,k;
double *A, *B, *C;
A = (double *)malloc(sizeof(double)*N*N);
B = (double *)malloc(sizeof(double)*N*N);
init_data(A, N*N);
for(i = 0; i < N; i++)
   for(j = 0; j < N; j++)
          B[i*N+j] = A[j*N+i];
// assume load A[j*N+i] and then store B[i*N+j]
output_data(B, N*N);
```

Assume that the starting address of array A is 0x20000 and array B is 0x40000. Assume N = 128. Please answer the following questions:

(1) Assuming that you have an AMD Phenom II processor, which has a 64KB, 2-way, 64-byte blocked L1 data cache, please estimate the cache miss rate.

C = ABS

64KB = 2 x 64 * S

S = 512

Offset = lg(64) = 6

Index = lg(512) = 9

Tag = 64-6-9 = 49 bits

2 way associative

A: 0010 0000 0100 0000 0000

B: 0100 0000 0000 0000 0000

	Address	Tag	Index	Hit/Miss
Load a[0 * 128+0]	0x20000	0x4	0x00	Miss
Store b[0 * 128+0]	0x40000	0x8	0x00	Miss
Load a[1*128+0]	0x20200	0x4	0x08	Miss
Store b[0 * 128+1]	0x40000	0x8	0x00	Hit
Load a[2 * 128+0]	0x20400	0x4	0x10	Miss
Store b[0 * 128+2]	0x40000	0x8	0x00	Hit
Load a[3*128+0]	0x20600	0x4	0x18	Miss

Store b[0 * 128+3]	0x40000	0x8	0x00	Hit
…	…	…	…	…
Load a[0 * 128+1]	0x20000	0x4	0x00	Hit
Store b[1 * 128+0]	0x40200	0x8	0x08	Miss
Load a[1*128+1]				
Store b[1 * 128+1]				

B misses 1/128, hits 127/128 the first 128; then B misses 1/8 times
B misses about 1/8(128x128) times
A misses all of the first 128, but hits all of the rest (128*128)
16384 total executions each of A and B
128/16384 + (1/8(16384))/16384 = .007 + .125 = .132 = About 13.2% Miss

(2) Continued from the previous question, how many of the misses are compulsory misses? How many of them are conflict misses?
All misses of A are compulsory misses, and 1/8 misses of B are compulsory misses are well. The rest are conflict misses.

128 + 2048/8 = 128 + 256 = 384 Compulsory Misses
2048 – 256 = 1792 Conflict Misses

2. 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	0x8	E
1	1	0xC	F
2	1	0x3	6
3	0	0xB	C

and the current page table is

	Valid	Physical page or in disk
0	1	0x5
1	1	0xD
2	0	Disk
3	1	0x6
4	1	0x9
5	1	0xB
6	0	Disk
7	1	0x4
8	0	0xE
9	0	Disk
10	1	0x3
11	1	0xC
12	1	0xF

Please identify how many TLB misses and page faults in the following address stream:

0x123D, 0x08B3, 0x365C, 0x871B, 0xBEE6, 0x3140, 0xC049

4kb pages = 12 bits page offset; 16-14 = 4 bit tag

I assume higher index is LRU so 0 is Most recently used and 3 is least in

Tag/page number and offset	TLB	Page Fault
0x1 23D	miss	Yes
0x0 8B3	miss	No
0x3 65C	miss	No
0x8 71B	Miss	yes
0xB EE6	miss	no
0x3 140	Hit	no
0xC 049	Miss	Yes

3. 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?

        (2^64B/4KB)*4B = 2^4PB
        2^4PB * ½ * 6 = 6^4PB

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

        C = ABS
        C = 4 x 8KB (from Page Size)
        C = 32KB

4. 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?

Processor runs at 2GHz

20% are loads

5% are stores

L1 I cache miss rate – 2%

1 cycle hit

50% evicted blocks are dirty

L1 D cache miss rate – 5%

1 cycle hit

L2 U cache miss rate – 20%

20 cycle hit

25% evicted blocks are dirty

L1 TLB miss rate – 2%

240 cycle penalty

1 cycle hit

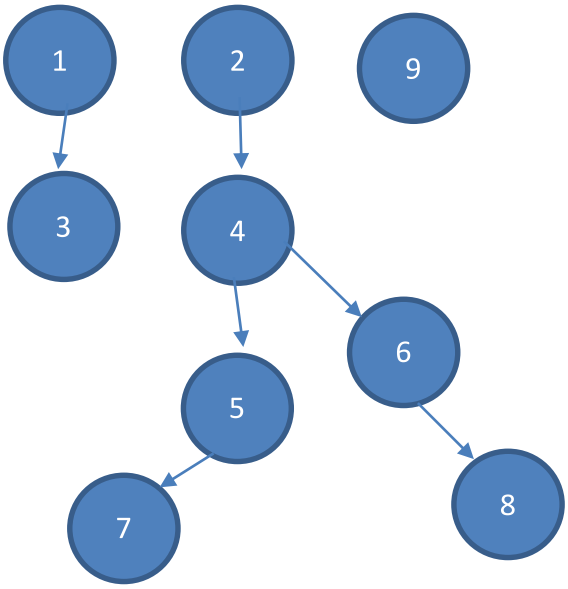
Main Memory hit time - 120 cycles

CPIaverage = 1+((.25)((.02)(240)+(.05)(1+(.5))(20+(.2)(1+(.25))(100))))+
((.02)(240)+1(.02)(20+(.2(1+(.25))(100)))) = 8.743 CPI

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 identifies the false dependencies (WAW and WAR) in the given code

False dependencies 1-2; 5-6; 7-9

(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?

lw $t1, 0($a0)	if	id	Ren	SCH	Exe	Mem	C			
addi $a0, $a0, 4	If	Id	Ren	Sch	Exe	C	C			
add $v0, $v0, $t1	If	Id	Ren	Sch	Sch	Sch	Exe	C		
bne $a0, $t0, LOOP	If	Id	Ren	Sch	Sch	Exe	C	C		
lw $t1, 0($a0)		If	Id	Ren	Sch	Exe	Mem	C		
addi $a0, $a0, 4		If	I	Ren	Sch	Sch	Exe	C		
add $v0, $v0, $t1		If	Id	Ren	Sch	Sch	Sch	Exe	C	
bne $a0, $t0, LOOP		If	Id	Ren	Sch	Sch	Sch	Exe	C	
sw $v0, 0($a1)			If	Id	Ren	Sch	Exe	Mem	C	

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. 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 cycle 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 execute these two threads on each processor?

	Thread A	Thread B
CPU I	(2+1+1)/4 = 1	(1+1+1)/4 = .75
CPU II	(2+1+1)/4 + 1	(1+1+1)/4 = .75

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

2 issue slots were wasted due to hazards