Name:_____

Computer Architecture
ELE 475 Final Exam
Fall 2012
David Wentzlaff

(Total Time = 180 Minutes)

Please read these instructions closely before you start work on this exam. This is an 180 minute timed exam. To take this exam, we recommend that you print out the exam and then take it on paper. Once you have completed your exam on paper, either type in your solutions into the text boxes for the exam or upload your answers as a pdf.

1. **You must use a fixed-width font for pipeline diagrams.** Before you type in or copy and paste your answers into the text box, make sure that you're using the 'Courier New' font, otherwise the formatting will not be preserved. You can select this font by clicking on the "Font Family" dropdown in the text box or in your text editor, and selecting the 'Courier New' font. Doing so will guarantee that columns line up in the results.

2. **Use a single underscore "_" to indicate blank spaces in the code sequence**. For example:

```
0: ADD R1, R2, R3    F  D  X  M  W
1: SUB R4, R5, R5    _  F  D  X  M  W
2: ADD R6, R7, R8    _  _  F  D  X  M  W
```

3. **If you are uploading your answers, please upload a pdf only.** Do not upload .doc, .docx, or other proprietary software files because your peers may not be able to open them during assessment.

After the exam has been completed on paper, you will have up until the submission deadline to upload or input your solutions onto the website. No answers should be changed, embellished, or expanded while typing. Be sure to show how you developed the solution, including how you derived the answer, but clearly state what your final answer is. Each problem and sub-problem should be entered separately.

You will have two weeks to complete the exam. The exam will close on **Monday, December 31 at 12:01 AM EST** (in the Princeton time zone) and the peer-assessment process will begin on **Tuesday, January 1 at 2:01 AM EST.** Peer assessments must be completed by **Wednesday, January 16 at 11:59 PM EST.** A rubric will be provided for you to use in reading and evaluating 5 exams from your peers in this course. You may then receive your own paper to assess and compare results with those from your peers. At the end of the peer evaluation period, you will receive your scores if you have submitted your evaluation of 5 other exams. You will receive a single score for each problem that represents the median of the scores given by your peers.

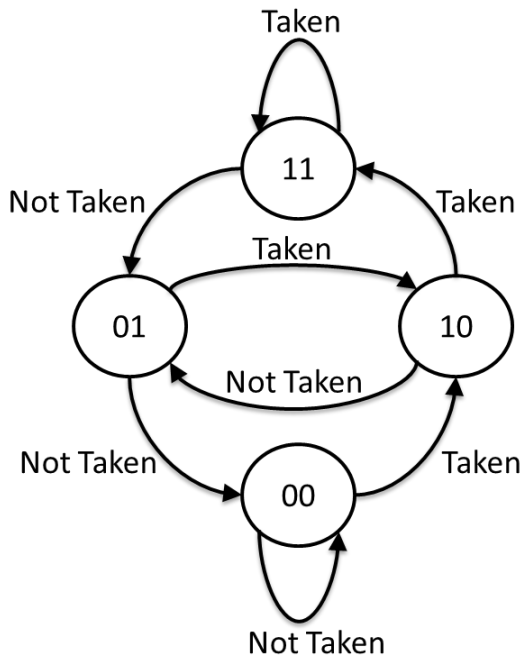
	Begins	Ends
Final Exam	Monday, December 17 at 12:01 AM EST	Monday, December 31 at 11:59 PM EST
Peer Evaluations	Tuesday, January 1 at 2:01 AM EST	Wednesday, January 16, 2013 at 11:59 PM EST

Name:_____

Problem 1) (5 Points) Briefly describe the purpose of a Branch Target Buffer (BTB).

Name:_____

Problem 2) For this problem, consider a processor with a 2048-entry Branch History Table (BHT), which is indexed by the low-ordered bits of the program counter (PC). The BHT implements a two-bit predictor with the state transition diagram shown below. The edges in the state transition diagram represent the branch outcomes that cause a transition. The predictor begins in the "00" state. The predictor predicts taken when it is in states "01", "10", and "11". The predictor predicts "not taken" when it is in state "00". The processor is a simple 5-stage pipeline with no branch delay slots. Assume that the result of the code is in registers R5 and R6. R5 and R6 contain zero at the beginning of exectuion. Assume that R3 contains a positive number 'n' at the beginning of execution of this loop.



[Figure showing state machine]

Problem 2 Code Sequence:

```
loop:
     LW R1, 0(R2)
     ANDI R4, R1, 1
b_1:
     BEQZ R4, l_3
     ADD R5, R5, R1
     J l_4
l_3:
     ADD R6, R6, R1
l_4:
     ADDI R2, R2, 4
     SUBI R3, R3, 1
b_2:
     BNEZ R3, loop
```

Name:_____

Part 2a) (3 points) What does this program compute? Conceptually, what does R5 and R6 contain when the loop exits?

Part 2b) (7 points) Assume that the initial value of R2 is 100. At address 100, there is an array of 32-bit integers with the following values in order of increasing memory location, [2, 1, 3, 4, 6, 6, 5, 7, 3, 2, 1]. Assume that at the beginning of execution that R3 contains 8. Fill in the following table by simulating the execution of the above loop.

Machine State					BHT State	Branch Behavior		Updated Values
PC	R2	R3	R5	R6	BHT Bits	Predicted Outcome	Actual Outcome	New BHT Bits
b_1	100	8	0	0	00	NT	T	10
b_2	104	7	0	2	00	NT	T	10
b_1								
b_2								
b_1								
b_2								
b_1								
b_2								
b_1								
b_2								
b_1								
b_2								
b_1								
b_2								
b_1								
b_2								

Name:_____

Part 2c) (2 points) What is the branch prediction accuracy for the above code execution for branch b_1? What is the branch prediction accuracy for branch b_2?

Part 2d) (5 points) The above architecture has a 1-cycle branch mispredict penalty and no penalty for correctly predicted branches. Jumps are always predicted correctly. Assuming that the above architecture is modified to support partial predication (conditional move with both move-if-zero and move-if-not-zero), using the prediction accuracy found for the execution of the code above, would it be fruitful to transform branch b_1 with partial predication?

Name:_____

Problem 3) (12 Points) Optimally schedule and bundle the following (sequential) code for a VLIW processor using the Equals (EQ) scheduling model. Unroll the code where appropriate and show prolog and epilog code. Perform code motion and register renaming where appropriate. The VLIW processor has 6 total functional units. The VLIW processor has two integer (ALU) units (X, Y), two multiply units (M, N), one load unit (L) and one store unit (S). Assume that branches and jumps must execute in the Y pipeline and that branches have no delay slots. The processor has full bypassing but no scoreboard (the VLIW processor does not stall on data dependencies). ALU instructions have a latency of 1 (result can be used next cycle), multiply instructions have a latency of 2 cycles, loads have a latency of 2 cycles, and stores have a latency of 2 cycles. Note that the loop has a fixed number of iterations. Also, achieve peak performance while minimizing instruction storage. As a secondary constraint, minimize register usage.

Problem 3 Code Sequence:

```
0: ADDIU R6, R0, 1232  // starting address
1: ADDIU R10, R0, 7  // loop counter
loop:
2: LW R7, 0(R6)
3: LW R9, 4(R6)
4: MUL R8, R7, R9
5: SW R8, 0(R6)
6: ADDIU R6, R6, 8
7: SUBI R10, R10, 1
8: BNEZ R10, loop
```

Name:_____

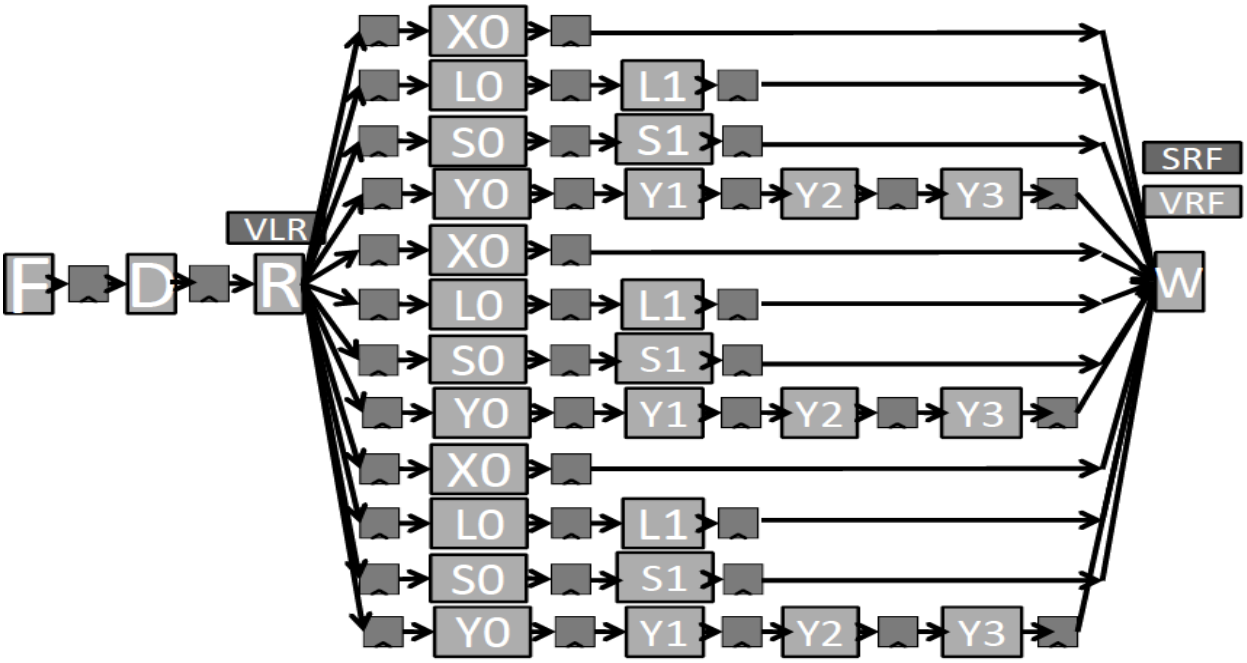
Problem 3 continued)

Name:_____

Problem 3 continued)

Name:_____

Problem 4) (12 Points) Draw the optimal pipeline diagram for the vector portion of the following code executing on a **3-lane** VMIPS vector processor shown below. The vector processor fetches instructions in-order, issues instructions in-order, and writes-back results out-of-order. Assume the processor can fetch one instruction per cycle and decode one instruction per cycle. Assume that the processor has **two read ports per lane** and **one write port per lane** for a total of 6 read ports and 3 write ports. Assume that the processor has a scoreboard to detect hazards. Assume that the processor can only bypass values **through the register file** and that values written to the register file in one cycle are not available until the next cycle. Assume that pipeline X can execute branches and ALU operations, pipeline L can excute loads, pipeline S can execute stores, and pipeline Y can execute all multiply operations. Loads have a latency of two cycles and ALU operations have a latency of one cycle. Branches are resolved in X0 and the machine has no branch delay slots and always predicts the fallthrough path. Multiply instructions have a latency of four cycles. Use the named pipeline stages in the figure for your pipeline diagram. The processor has a dedicated register read stage, denoted as "R" in the figure.



[Figure showing Three Lane VMIPS Processor Pipeline]

Problem 4 Code Sequence:

```
C Code:
for (i = 0; i < 6; i++)
{
   a[i] = (b[i] * c[i]);
   d[i] = (b[i] + c[i]);
}
```

```
VMIPS:
ADDI R9, R0, 6
MTC1 VLR, R9
LV V1, R10 //start diagram here
LV V2, R11
MULVV.D V3, V1, V2
ADDVV.D V4, V1, V2
SV R7, V3
SV R6, V4
```

Name:_____

Problem 4 continued)

Name:_____

Problem 4 continued)

Name:_____

Problem 5) (5 Points) What problem does vector stripmining solve? How is the Vector Length Register (VLR) involved with stripmining?

Name:_____

Problem 6) (12 Points) Show for each cache line and per cache what state it is in on every cycle assuming four processors executing the code as interleaved below. Assume a 128-byte cache line (block size). Assume all cores contain a direct mapped data cache of size 2KB. You do not need to show lines which are invalid in the cache, but you should remove them from the table when the lines transition to the Invalid state. Assume that the caches implement a snoopy, bus-based, cache coherence protocol implementing the MESI protocol. MESI stands for the four states, Modified, Exclusive, Shared, and Invalid. Assume all lines are Invalid in all caches at the beginning of execution. Be sure to duplicate the information from time step to following time step if the data stays valid in the cache and show the state that the data is in.

Time	Processor 1 Code	Processor 2 Code	Processor 3 Code	Processor 4 Code
1	`LW R6, 128(R0)`			
2			`SW R0, 264(R0)`	
3				`LW R7, 132(R0)`
4				`SW R10, 128(R0)`
5		`SW R0, 384(R0)`		
6		`LW R11, 144(R0)`		
7		`LW R12, 388(R0)`		
8			`LW R9, 128(R0)`	
9			`LW R10, 2180(R0)`	
10				`LW R15, 512(R0)`
11				`SW R7, 516(R0)`
12	`SW R6, 384(R0)`			
13	`LW R9, 228(R0)`			
14			`SW R10, 200(R0)`	

Please complete the table on the next two pages.

Name:_____

Problem 6 continued)

Time	Processor 1 Cache		Processor 2 Cache		Processor 3 Cache		Processor 4 Cache	
	Line Address	Line State	Line Address	Line State	Line Address	Line State	Line Address	Line State
1	128	E						
2								
3								
4								
5								
6								
7								

Name:_____

Problem 6 continued)

Time	Processor 1 Cache		Processor 2 Cache		Processor 3 Cache		Processor 4 Cache	
	Line Address	Line State	Line Address	Line State	Line Address	Line State	Line Address	Line State
8								
9								
10								
11								
12								
13								
14								

Name:_____

Problem 7) For this problem, you are designing a processor with a 64KB data cache that is four-way set associative with a 128-byte block size and a not-most recently used (NMRU) replacement policy. This processor has a virtual memory system with a software managed TLB. Virtual addresses on the machine are 32-bits and physical addresses are 32-bits. The TLB contains 16 entries and is 8-way set associative. Assume that the architecture flushes the TLB and cache on process swap. Assume that the page size of the machine is 16KB. Assume that each page table entry requires a valid bit and a dirty bit. Likewise, each TLB entry contains a valid bit and a dirty bit. Assume that the architecture is byte addressable. Assume that the cache is virtually indexed and physically tagged. Assume that the TLB uses 3 bits to implement a NMRU replacement policy.

Part 7a) (4 Points) For the above described processor, fill in the following table of bit widths and sizes:

Data Cache		**TLB**	
Tag width	bits	Page Offset width	bits
Index width	bits	Virtual Page Number width	bits
Offset width	bits	Physical Page Number width	bits
Data Field Size	bytes	Total TLB data payload width (not including tag, valid bit, or NMRU)	bits
		TLB tag field width	bits
		TLB index field width	bits

Part 7b) (7 Points) In the space below, neatly draw labeled diagrams of the cache and TLB showing addresses leading into them. Be sure to show (label) the widths of buses and which bits are used for different portions of indexing and matching. Show where the data payload is output from the cache and TLB and be sure to show the logic for a miss in the cache and TLB. For this part, assume that the processor can access only byte aligned, byte sized words. You **do not** need to show the **write path** of the TLB or Cache. You need to show everything required to construct a behavioral simulation of the Cache and TLB, for example, bitwidths for indexing lines into decoders, number of bytes per data entry, valid bits, comparitors, different banks for associativity, etc.

Name:_____

Part 7b continued)

Name:_____

Part 7b continued)

Name:_____

Part 7c) (4 Points) Even though the above processor has a software managed TLB, it is desirable for the operating system to use a multi-level page table in order to save space. Design a multi-level page table structure which fits page table entries naturally into a page and favors having the leaf page table level full of page table entries. How many levels is this page table structure? Draw a figure showing which and how many bits are used for each purpose of a page table walk. Label the bits and widths.

Name:_____

Problem 8) (5 Points) Write a sequence of code showing false sharing. Assume that the code is executing on a shared memory multiprocessor system with caches and denote what code executes on each processor and any assumptions about cache block size.

Name:_____

Problem 9) (5 Points) Two threads are executing on two independent processors. The data stored at address ‘p’ is initalized to the value 5 and the data stored at address ‘q’ is initialized to the value 6 before the threads begin executing. Note that `SW R0, 0(q)` stores the value zero at address ‘q’. After both threads complete execution, is the state where the data stored at address ‘p’ contains the value 31 and the data stored at address ‘q’ contains the value 10 a sequentially consistent execution?

Thread 1	Thread 2

`LW R1, 0(q)`
`ADDI R1, R1, 1`
`LW R2, 0(p)`
`ADD R2, R2, 5`
`SW R1, 0(p)`
`SW R2, 0(q)`
	`ADDI R3, R0, 30`
	`SW R3, 0(p)`
	`SW R3, 0(q)`

Name:_____

Problem 10) (7 Points) The following code is designed to implement a producer-consumer relationship between two independent threads which execute on different processors. These threads share memory and the memory system is sequentially consistent. Assume that the variables `ring_buffer`, `head_index`, `tail_index`, and `number_in_buffer` are shared between the threads. Assume that statements in the code provided below execute in order, but any given statement may not be atomic. The functionality of `SendData` puts one value into the communication buffer and `ReceiveData` removes one value from the buffer. Assume that the first thread only ever calls `SendData` and the second thread only ever calls `ReceiveData`. Assume that there is a library which implements a mutual exclusion lock (mutex) which have two functions `AcquireLock(int * lock)` and `ReleaseLock(int * lock)`. Both `AcquireLock` and `ReleaseLock` take a parameter which is the address of a lock variable. Also, the lock library has a function `InitializeLock(int * lock)` which initializes a lock in the unlocked state when given a parameter which is the address of a lock. For proper operation, does the following code need locking? If so, add the minimal `AcquireLock`, `ReleaseLock`, `InitializeLock,` and lock variable instantiations as needed. Be sure to only hold the lock for the only the minimal time as is necessary. Do not reorder the code below. Feel free to edit in-place, but please be clear of changes.

```
#define BUFFER_SIZE 4
int ring_buffer[BUFFER_SIZE];
int head_index = 0;
int tail_index = 0;
int number_in_buffer = 0;
void SendData(int data)
{
   while (number_in_buffer < BUFFER_SIZE) {
   }
   number_in_buffer = number_in_buffer + 1;
   ring_buffer[tail_index] = data;
   tail_index = (tail_index + 1) % BUFFER_SIZE;
}
int ReceiveData()
{
   int the_data;
   while (number_in_buffer <= 0) {
   }
   number_in_buffer = number_in_buffer – 1;
   the_data = ring_buffer[head_index];
   head_index = (head_index + 1) % BUFFER_SIZE;
   return the_data;
}
int main(void)
{
   LaunchThreads();
   return 0;
}
```

Name:_____

Problem 10 continued)

Name:_____

Problem 11) (5 Points) Which network has a higher minimum bisection bandwidth and why? A 2-ary 3-cube where each link is 16-bits wide, with 8-bits in each direction and the links are clocked at 100MHz, or a 4-node shared multi-drop bus which is 64-bits wide and clocks at 200MHz.

END OF EXAM